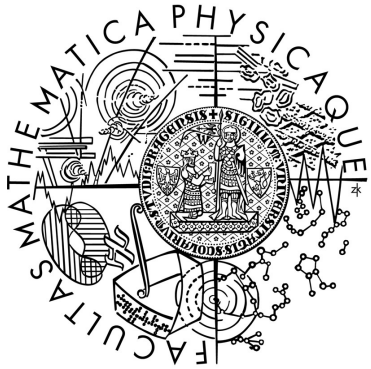
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Ľubomír Dáni

Elektronický obchod jako služba

Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. David Bednárek Ph.D.

Studijní program: Informatika

Studijní obor: Programování

Praha 2011

Chcel by som sa poďakovať pánovi RNDr. Davidovi Bednárkovi Ph.D. za jeho ochotu a rady pri vypracovávaní tejto bakalárskej práce.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 05.08.2011 podpis

Název práce: Elektronický obchod jako služba

Autor: Ľubomír Dáni

Katedra / Ústav: Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. David Bednárek Ph.D., KSI

Abstrakt: Cieľom tejto práce je navrhnúť a implementovať systém aplikácii, ktoré umožňujú koncovému užívateľovi pomocou webového prehliadača vytvárať a spravovať elektronické obchody. Webový obchod okrem klasických bežných b2c operácií umožňuje správu zamestnancov, úpravu vzhľadu obchodu a zdieľanie šablón vzhľadu medzi obchodmi. Systém sa skladá zo správy obchodov, prezentácie obchodov a administrácie obchodov. Správa obchodov dovoľuje vytvoriť nový obchod, zablokovať jeho funkčnosť, prípadne odstrániť kompletne obchod. V prezentácii obchodu zaisťuje registráciu jeho klientov, tvorbu objednávok a ich prehľad. Administrácia obchodu nám ponúka možnosť meniť vzhľad prezentácie, jednoduchú logistiku produktov, ich objednávok a klientov.

Klíčová slova: e-shop, služba, B2B

Title: E-shop as a service

Author: Ľubomír Dáni

Department: Department of Software Engineering

Supervisor: RNDr. David Bednárek Ph.D., KSI

Abstract: The goal of this thesis is to develop and implement the system of applications, which would enable the user to create and manage e-shops via the web browser. In addition to the classical B2C operations web e-shop enables also the management of employees, modification of e-shop appearance and sharing of appearance templates between different e-shops. The system consists of the management, presentation and administration layer. The management layer enables to create, block or delete e-shop. The presentation layer enables the registration of clients and the creation of orders. The administration layer enables the modification of the appearance, simple logistics of products, orders and clients.

Keywords: e-shop, service, B2B

Obsah

Úvod / Předmluva

S rozvojom internetu a nárastom počtu internetovo gramotných ľudí sa vytvoril segment poskytujúci softvér ako službu. Ten má veľkú výhodu pre nízke náklady zo strany klienta.

Spomenutý trend neobišiel ani elektronické obchody, ktoré sú už bežným nástrojom komunikácie medzi predajcom a zákazníkom. Na trhu sú dostupné mnohé riešenia obchodov, ktoré sa líšia cenou, náročnosťou na infraštruktúru, obsluhou, zameraním a funkciami. Medzi najdostupnejšie patrí práve poskytovanie elektronického obchodu ako služby. No samotných systémov pre ich poskytovateľov veľa nie je.

Služby majú za cieľ pokryť v danom sektore najviac potenciálnych klientov, ktorých požiadavky bývajú rôznorodé, čo znamená väčšie nároky na funkčnosť oproti softvéru na zákazku. S množstvom funkcií pri softvéri je spojený aj dlhší a nákladnejší vývoj, ktorý zrejme odrádza mnohých vývojárov a preto sa softvér ako služba stali doménou najmä veľkých spoločností.

Medzi kritické požiadavky funkčnosti aplikácií, ktoré odberateľovi poskytujú nejakú formu prezentácie na internete, je možnosť budovať vlastnú identitu, ktorá ho má odlíšiť od ostatných užívateľov služby, resp. pomôcť mu v konkurenčnom boji na trhu. Preto je vhodné, aby riešenie malo šancu meniť vzhľad prezentovaného obchodu, nastavovať obsahové a logistické parametre bez nutnosti zásahu poskytovateľa.

Náplňou práce je vytvorenie systému aplikácii, ktorých výsledkom je poskytovať elektronický obchod ako službu. Tento celok bol nazvaný Generátor elektronických obchodov.

Cieľom nasledujúcej kapitoly je uviesť čitateľa do problematiky elektronických obchodov a softvéru ako služby a oboznámenie sa s niektorými konkurenčnými nástrojmi, ich výhodami a nevýhodami a popis princípov a cieľov Generátora elektronických obchodov. Ďalšia kapitola rozoberie vlastnosti aplikácie a práce s ňou z pohľadu bežných užívateľov. Štvrtá kapitola sa zaoberá samotnou implementáciou riešenia. Predstavuje technológie, ktoré sú použité a postupy, ako aplikáciu ďalej rozširovať a prispôsobiť tak jednotlivé funkcie špecifickým požiadavkám. Samotná problematika ponúka niekoľko základných otázok a v konečnom dôsledku aj odpovedí.

1. Motivácia

Nárast užívateľov internetu v posledných rokoch badať vo svete a aj v Českej republike (Obrázok č.1). Užívatelia si postupne zvyšujú aj mieru svojej internetovej gramotnosti (Obrázok č. 2). Teda rozširujú svoje vedomosti o rôznych službách, ktoré im celosvetová sieť ponúka. Mnohí z nich takto objavili nesporné výhody v rýchlom prístupe k informáciám.

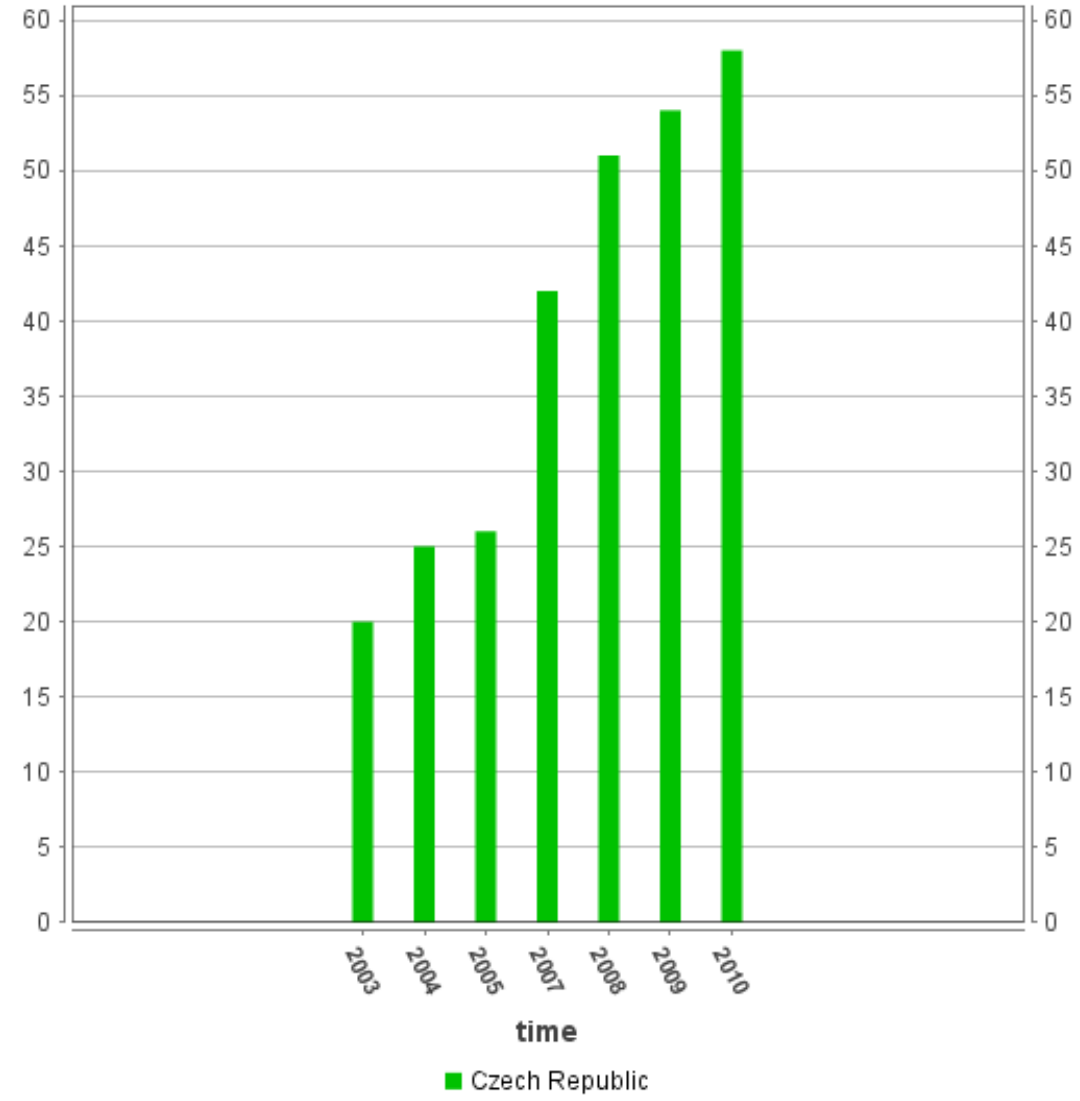
Základným nosičom týchto informácii je www stránka vo formáte html. Kedysi sa jednalo len o statické stránky obsahujúce text, odkazy na iné stránky a obrázky. Následným vývojom technológii stránky začali obsahovať multimediálne prvky ako videá, aktívne prvky ako formuláre, rôzne skripty a štýly určujúce vzhľad stránky a formátovanie textu [5]. Toto umožnilo prezentáciu produktov prostredníctvom webu, čiže možnosť rýchlo a lacno osloviť nových zákazníkov nezávisle na lokácii predajcu. Vzniká tak nový fenomén popri klasickom trhu – elektronické obchodovanie. V skratke ide o výmenu kapitálu, tovarov a služieb sprostredkovanú pomocou internetu[1].

Rozoznávame viacero druhov elektronického obchodovania podľa povahy transakcie. Ak prebieha medzi dvoma subjektmi podnikovej sféry, hovoríme o type medzi podnikom a podnikom (business to business – B2B). Jedná sa o predaj, kúpu spracovateľských produktov alebo služieb, ktoré nemajú za cieľ oslovovať koncového konzumenta produktu. Ďalší typ je medzi podnikom a individuálnym klientom (business to customer - B2C), v ktorom ide najmä o predaj spotrebného tovaru. Teda ide o predaj koncového produktu. Najznámejším preborníkom na svete v tejto oblasti je elektronický obchod amazon.com. U nás sa dlhodobo darí v tejto oblasti podnikom mironet.cz, alza.cz alebo mall.cz. Hodnoty transakcií B2C sú rádovo nižšie ako pri B2B, no aj tak ide o najdiskutovanejší a najčastejší druh vo svete elektronického obchodovania. Pre úplnosť uvediem v Čechách aj vo svete udomácnený typ, ktorým je výmena medzi dvoma indivíduami (customer to customer – C2C). Takéto transakcie sú najviac badateľné na dobre známych projektoch ebay.com alebo aukro.cz, ktoré vytvorili digitálne obdoby aukčných siení, umožňujúce dražiť i dávať veci do dražby skoro každému účastníkovi internetu[1].

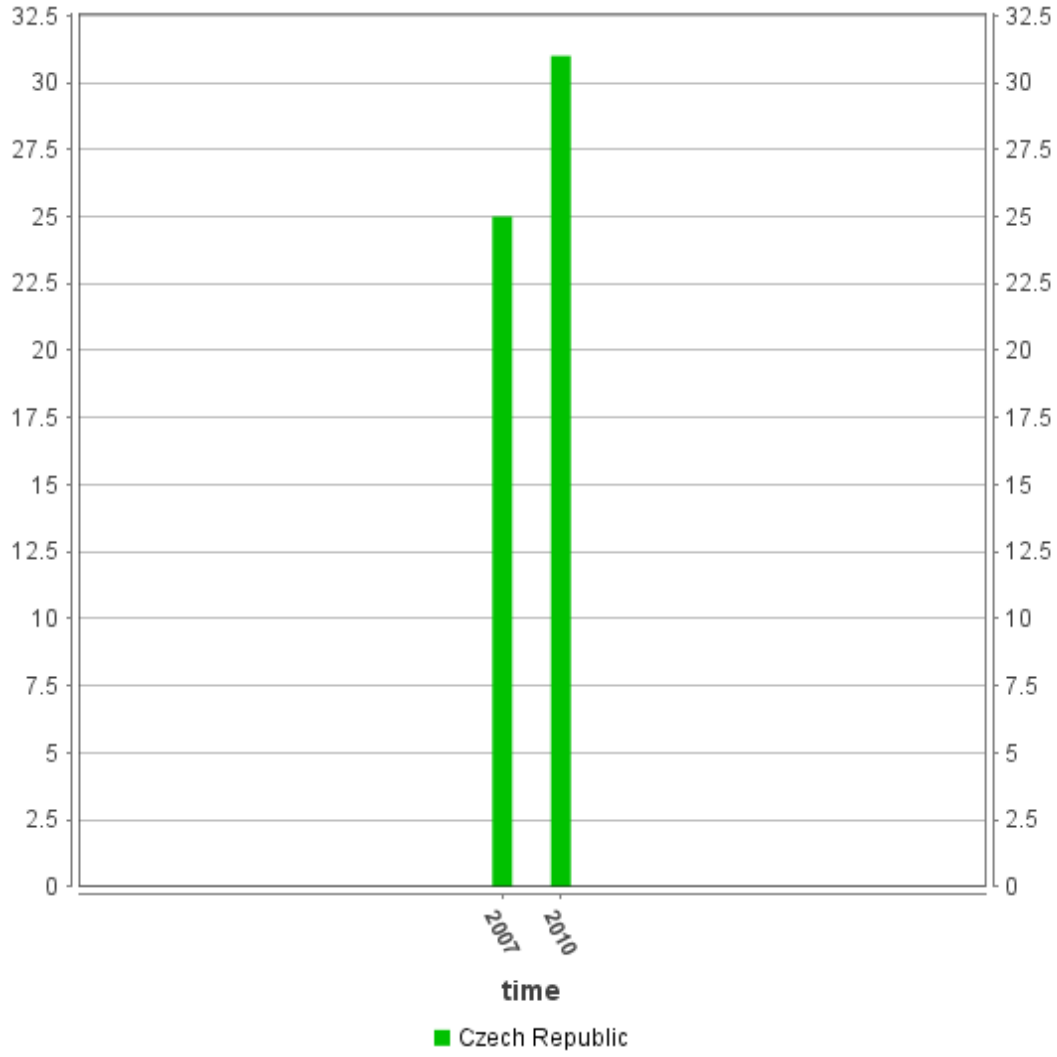
Vlajkovými loďami B2C modelu sú elektronické obchody. Je to akási digitálna obdoba kamenného obchodu umožňujúca návštevníkovi pomocou webového prehliadača prezerať tovar, simulovať nákupný košík a uzavrieť nákupnú transakciu medzi predajcom a zákazníkom. Dnes majú predajcovia k dispozícii rôzne krabicové hotové elektronické obchody (open-source riešenia: Magento, osCommerce), krabicové nástroje pre ich tvorbu (vhodné najmä pre vývojárov: Microsoft Frontpage, Macromedia Dreamweaver) alebo možnosť dať si vyrobiť obchod na mieru. Každé zo spomenutých riešení núti prevádzkovateľa zaoberať sa alebo explicitne investovať do rozbehnutia a prevádzkovania obchodu aj po technickej stránke, čo môže mnohých ľudí, hoc uvedomelých o existencii internetu, odradiť od rozšírenia svojho fyzického podnikania do digitálneho sveta.

Softvér ako služba nám často ukazuje kombináciu viacerých modelov elektronického obchodovania v jednom systéme aplikácii. Aj to nám napovedá o obtiažnosti vývoja týchto softvérových riešení zmienenej v úvode, a z toho vychádzajúceho nedostatku open-source aplikácii, najmä generátorov elektronických obchodov. Samozrejme druhou stránkou mince, ta pozitívna, je širšie spektrum

odberateľov, keďže ich odbremeňuje v najväčšej možnej miere od starostí sveta informačných technológií spojených s používanou aplikáciou, ktoré ležia na pleciach poskytovateľa služby. Tým umožňuje odberateľovi plne sa sústrediť na využívanie funkcií pre prospech svojho biznis modelu.



(Obrázok č. 1. – eurostat vývoj internetovej populácie)



(Obrázok č. 2. – eurostat vývoj počítačovej gramotnosti)

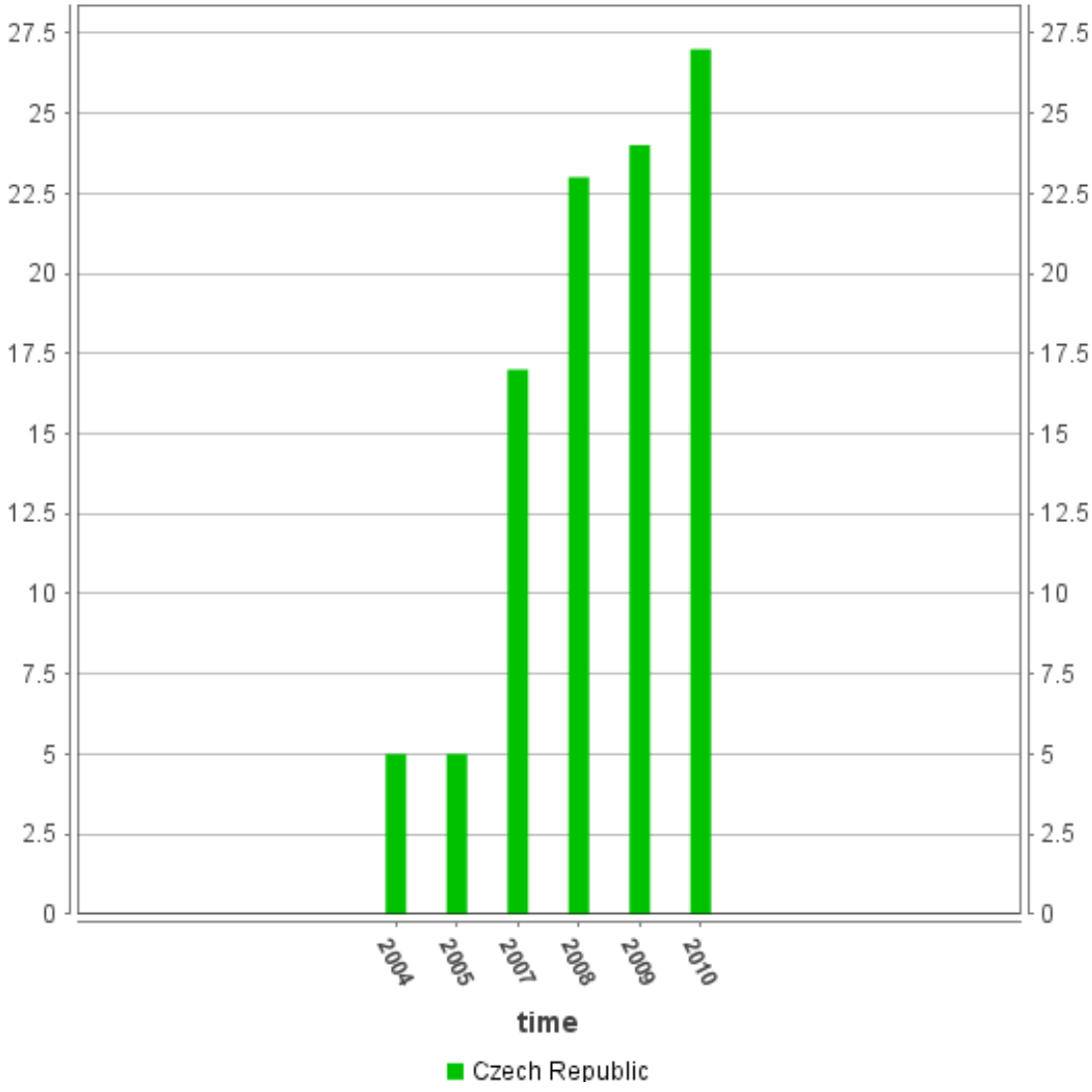
1.1. Elektronické obchody

Vývojom našej spoločnosti v oblasti obchodu dochádzalo aj k rozvoju platobných spôsobov. Akcelerátorom boli križovatky obchodných ciest a miest, kde sa koncentrovala populácia. Obchodníci sa so zvyšujúcou konkurenciou museli začať špecializovať, odlišovať sa a zamerať sa na prezentovanie produktov. To prirodzene viedlo k vzniku priestorov - obchodov, ktoré si klienti s nimi asociovali. V nich si mohli tovar prezrieť a prípadne kúpiť.

Preto aj v rámci celosvetovej siete začali mať niektoré webové aplikácie s rozvojom technológii funkcie napodobňujúce dobre známe akcie z fyzických - „kamenných“ obchodov. Medzi ne patrí predvádzanie tovaru, alebo služieb. V „kamenných“ obchodoch je tovar uložený v regáloch podľa špecifických vlastností. Obdoba v digitálnej podobe je často náhľad produktu obsahujúci fotografiu a text nesúci popisujúce informácie. Produkty sú v relácii podľa vlastností s kategóriami, ktoré možno prirovnať k regálom. Namiesto mechanického presunutia požadovaného objektu rukou do košíku sa na webových stránkach stretávame s odkazom alebo iným aktívnym prvkom, ktorý sprostredkuje pridanie do virtuálneho košíka. Ten je zväčša reprezentovaný na samostatne dostupnej stránke, ktorá je bránou k pokladni

tvorenej formulármi so vstupmi na vytvorenie objednávky. Existujú rôzne spôsoby platby za dodaný tovar alebo služby. V Českej republike je stále najpopulárnejšia platba na dobierku, v závese je platba v hotovosti pri výdaji. Medzi elektronické úhrady, ktoré má zabudované mnoho systémov, patrí platba kreditnou kartou, prevodom, alebo pomocou platobných agregátorov (služby typu PayPal) [7]. Webovú aplikáciu so spomenutými črtami budeme nazývať elektronickým obchodom.

Samozrejme nové obchodné rozhranie prinieslo lacné, jednoducho aplikovateľné nástroje marketingu, v podobe selektívnych zliav, hromadného rozosielania noviniek, komentárov k produktom, archívom užívateľských transakcií, či komunikáciou s klientmi pomocou sociálnych sietí. Aj vďaka tomu sa zvyšuje percento českej internetovej populácie, ktoré okúsilo túto formu obchodnej výmeny.



(Obrázok 3 – eurostat vývoj internetovej populácie, ktorá využila e-commerce)

Doteraz sme sa letmo oboznámili s časťou problematiky, ktorá sa zaoberala najmä interakciou medzi obchodníkom a klientom. No i v „kamennom“ svete to tvorí len vrchnú vrstvu celého obchodu. Aby sme sa k nej dostali, musí niekto najprv prebrať tovar a následne ho vyložiť. Ďalej sa starať o jeho predajnosť a nakoniec obslúžiť zákazníka pri pokladni. V digitálnom svete nám toto všetko zabezpečuje

administrátorské rozhranie aplikácie. To sa rozsahom funkcií bude u každého obchodu líšiť tak, aby reflektovalo zameranie, logistiku, podporované platobné systémy a iné potreby.

1.2. Softvér ako služba

Marketing ako nevyhnutná súčasť všetkého napomohla k vzniku termínu „cloud computing“ zahŕňajúceho vo všeobecnosti čokoľvek, čo poskytuje služby po sieti. Odlišuje sa od klasického server - klient modelu dodaním služby zo servera odberateľovi bez nutnosti inštalácie špeciálnej klientskej aplikácie. Na správu a prácu mu postačuje ako platforma webový prehliadač alebo iný klient nezávislý na samotnej službe. Dôsledkom je oddelenie užívateľa od technologickej infraštruktúry a aktualizácii systému. Tiež tradičný spôsob platby badaný pri hostovacích službách je iný. Spoplatnené býva to, čo v danom okamžiku odberateľ potrebuje. Jedná sa buď o dátové toky, spotreba miesta, alebo jednotlivé funkcie aplikácie. Táto flexibilita mu znižuje náklady a tiež garantuje chod využívaných služieb pri nepredpokladanej spotrebe. Ako príklad si predstavme aplikáciu bežiacu na bežnom hoste s fixovanou veľkosťou dátového úložiska. V prípade potreby uloženia dát nad hranicu je potrebné zmeniť využívaný host a aplikáciu pripraviť na prípadné stavy plynúce z neuložených dát. Avšak pri využívaní „cloud computing“ služby dátového úložiska s jednotkovou cenou za využitý priestor takáto situácia nenastane (ak nebude prekročená garantovaná hranica funkčnosti služby).[8]

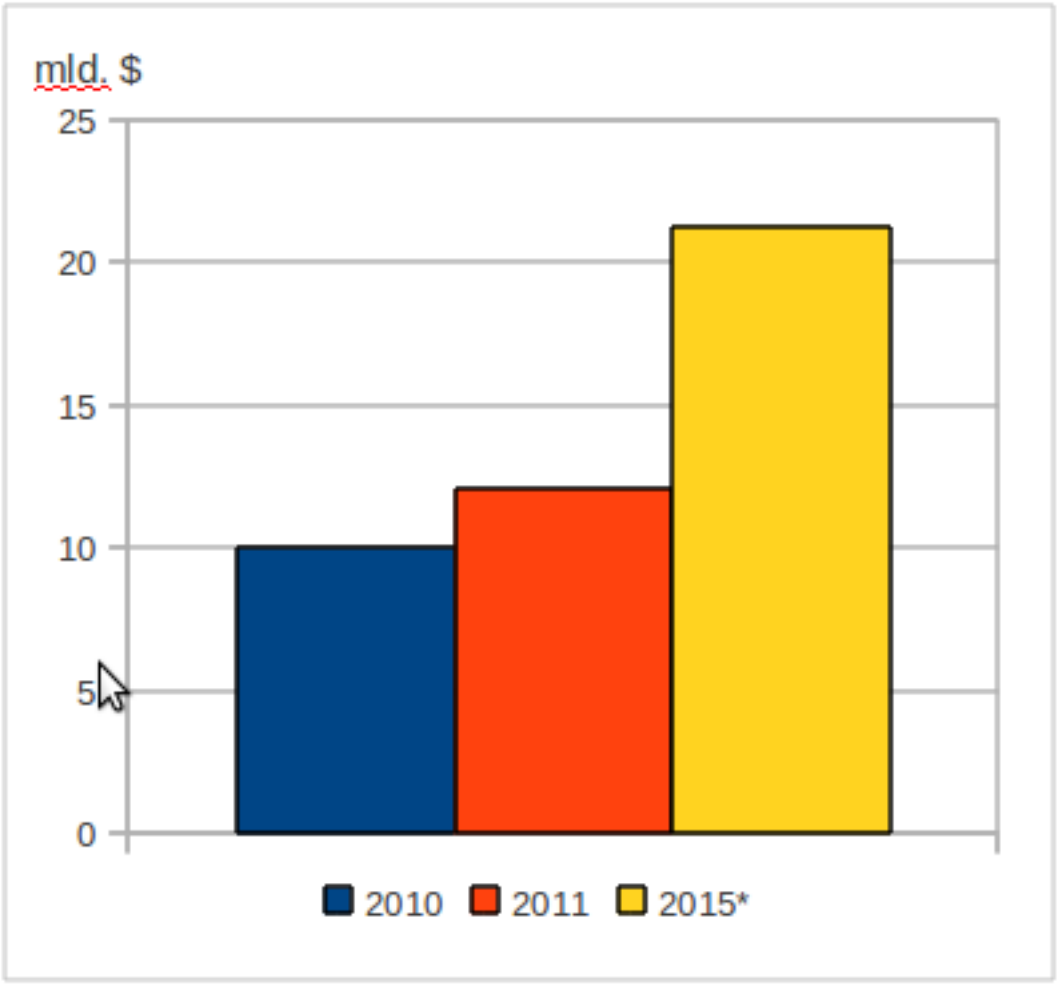
V praxi sa stretávame buď s verejnými „cloud“ modelmi, ktoré sú dostupné každému účastníkovi internetu, alebo so súkromnými, ktoré bežia v uzavretých sieťach, prípadne vyhradenými pre niekoľko vybratých užívateľov internetu.[9]

Spôsob, akým sú nasadené tieto moduly, nie je jediným faktorom, podľa ktorého ich možno kategorizovať. To, čo služby distribuujú, je pre potenciálneho konzumenta zásadné. Poskytovateľ IAAS „Infraštruktúra ako služba“ (Infrastructure as a Service – IAAS) nám garantuje použitie infraštruktúry, teda zaväzuje sa starať o hardware. Často sa v tejto oblasti ponúka virtualizácia platformy, či úložiská v dátových centrách. Známy predstavitelia týchto služieb sú Amazon Web Service, Windows Azure, Rackspace.

Skratka PAAS znamená „platforma ako služba“ (Platform as a Service – PAAS). Ponúka nielen sieťové služby, ale aj softvérovú vrstvu s vývojárskymi nástrojmi (rôzne IDE, API) v kombinácii s prostredím, kde nami vytvorené a spravované riešenie bude bežať. Priekopníkmi sú GoogleApps alebo Force.com (SalesForce.com). V prípade nespokojnosti so službou po dlhšom časovom období ale nastáva veľký problém. Systém, ktorý konkrétna organizácia vytvorila pomocou danej platformy, je v podstate neprenosný. Neexistujú žiadne štandardy,a preto si treba dopredu rozmyslieť, či je ponúkané riešenie vhodné. To znamená, či ponúkané benefity vyvážia vysokú mieru závislosti, alebo prípadnú nákladnú migráciu projektu.[8]

Širšia skupina ako predchádzajúce je SAAS - softvér ako služba (Sofware as a Service – SAAS). V tomto modeli obhospodáruje poskytovateľ všetky technické záležitosti z pohľadu hardvéru aj softvéru. Z centrálneho prvku systému distribuuje aplikáciu po sieti. Zákazník má dostupný front-end, ktorý prezentuje dáta, alebo ponúka prácu s objednanými nástrojmi a funkciami aplikácie. Mnoho organizácii presúva svoje krabicové riešenia do tejto formy. Výhody, ktoré im z toho plynú, sú zjavné pre obe strany. Majiteľ autorských práv aplikácie znižuje takto riziko ich

zneužitia, keďže užívateľ nemá k dispozícii samotnú aplikáciu. Tiež môže nezávisle na klientovi aktualizovať svoje riešenie, čím zlepšuje kvalitu riešenia. Na opačnej strane odberateľ využíva, len to, čo potrebuje, v časovom rozmedzí, keď to potrebuje. Na prácu mu postačuje webový prehliadač, nič nemusí inštalovať. Tým sa mu znižujú náklady, ako na ľudskú pracovnú silu, tak na licenčné poplatky za prerekvizity nutné k rozbehnutiu krabicovej verzie. Samozrejme aj tu nastáva určitá forma závislosti, ktorá je ale podobná ako pri klasickom softvéri. Financovanie sa presunulo z nákupu licencií, ktoré aj tak nútilo klienta si kúpiť s každým nasledujúcim rokom novšiu verziu aplikácie, na pravidelné poplatky s príplatkami za nadštandardné funkcie. Softvér ako služba nie je doménou len prameho financovania. Veľa služieb je užívateľom prístupných zdarma a na svoj chod si zarábajú nepriamo pomocou reklamy [10]. Narastajúcu obľúbenosť SAAS demonštrujú aj vzostupné hodnoty objemu tržieb v danom sektore. Viď Obrázok 4. [11]



(Obrázok 4 – vývoj trhu SAAS vo svete)

Už dlhšie obdobie kraľujú medzi SaaS systémy riadiace vzťahy so zákazníkmi (Customer relationship management – CRM), ktoré majú na svedomí veľké spoločnosti ako Oracle, Microsoft a iné. Nasledované sú riešeniami pokrývajúce prácu s ľudskými zdrojmi, účtovníctvom. Po týchto službách určených hlavne korporáciám na ich chod, si ukrajujú z koláča SaaS trhu aplikácie určené na elektronické obchodovanie (e-commerce). Do tejto sekcie spadá aj Generátor

elektronických obchodov. Medzi iné obľúbené služby, ktoré sú postavené na modeli SaaS, patria rôzni weboví e-mail klienti ako gmail.com, komunikátory (meebo.com) a rada ďalších aplikácii.[12]

Medzi nevýhody väčšiny „cloud“ systémov patrí nesúlad medzi justičnými systémami prevádzkovateľa a odberateľa alebo stále pretrvávajúca nedôvera klientov k novej, neotestovanej technológii.

1.3. Generátor elektronických obchodov

Existuje veľa open source elektronických obchodov, no väčšina predajcov tovarov nemá dostatočné znalosti v IT sektore a je pre nich zložité, ba priam niekedy nedosiahnuteľné nájsť pre nich vhodný open source elektronický obchod. Ak by sa im to podarilo, museli by nájsť vhodnú web hostingovú službu, na ktorej by bola aplikácia spustiteľná. Následne ju nakonfigurovať a až potom s ňou pracovať na biznis úrovni. Z pohľadu obchodníka to znamená rozšíriť svoj tím o človeka zaoberajúceho sa IT, čo pre malé spoločnosti predstavuje zbytočnú záťaž.

Vhodným riešením v takýchto prípadoch je nájsť elektronický obchod distribuovaný cez internetovú sieť ako službu. Pre obchodníka to bude znamenať registráciu – vytvorenie účtu na stránkach poskytovateľa (front-end generátora elektronických obchodov) a akceptovanie podmienok. Je tu prítomný obdobný postup, ktorý pozná z iných ním využívaných služieb ako sociálne siete, e-mail klienti. Následne mu tento centrálny prvok ponúkne prístup k aplikácii elektronického obchodu (v prípade, ak už bola „vytvorená“, inak nutný medzikrok generovania), ktorá preň predstavuje nástroj na komunikáciu s jeho zákazníkmi (viď sekcia 2.2.). Pre predstavu životného cyklu tohto biznis modelu nám slúži obrázok č. 5.

Povaha vzťahu medzi poskytovateľom služby a odberateľom sa v rámci elektronického obchodovania kvalifikuje ako B2B. Popri tom každý distribuovaný obchod vytvára vzťah medzi prenajímateľom a jeho zákazníkmi popísaný modelom B2C. V prípade, ak by táto B2C časť systému implementovala len základné funkčné požiadavky na elektronický obchod, tak by potenciálny nákupcovia tovarov mohli nachádzať obchody vygenerované rovnakým systémom ponúkajúce rôzny tovar, s rôznymi predajcami, ale rovnakým rozložením nosných prvkov. Citlivo reagujúci nákupca by mal problém vytvárať si asociácie k prenajímateľovi, a dokonca by to u mnohých vyvolalo podozrenie na niečo nekalé. Jednotliví predajcovia by tým strácali na dôveryhodnosti, ktorá je vo svete bez kontaktného predaja kľúčová.[1] Riešenie by tým stratilo na atraktivite. Preto distribuovaný webový obchod popri klasických funkciách, alebo centrálny prvok, musí ponúkať možnosť vytvorenia vlastnej identity vzhľadu.

Systém spĺňajúci opísané vlastnosti pomenujeme „Generátor elektronických obchodov“ a jeho vytvorenie je cieľom tejto práce. Jedná sa o riešenie vhodné pre poskytovateľov softvéru ako služby, takže obsahuje aj popis minimálne nutnej technickej infraštruktúry a nasadenia systému do prevádzky (viď príloha A - Inštalácia systému). No hlavným ťažiskom práce je vytvorenie aplikačnej vrstvy pozostávajúcej z centrálneho prvku a distribuovaného elektronického obchodu. To so sebou prináša rozobratie problémov, náčrt alternatívnych riešení užívateľského rozhrania (viď časť užívateľská dokumentácia), návrhu systému a samotnej implementácie (viď časť implementácia).

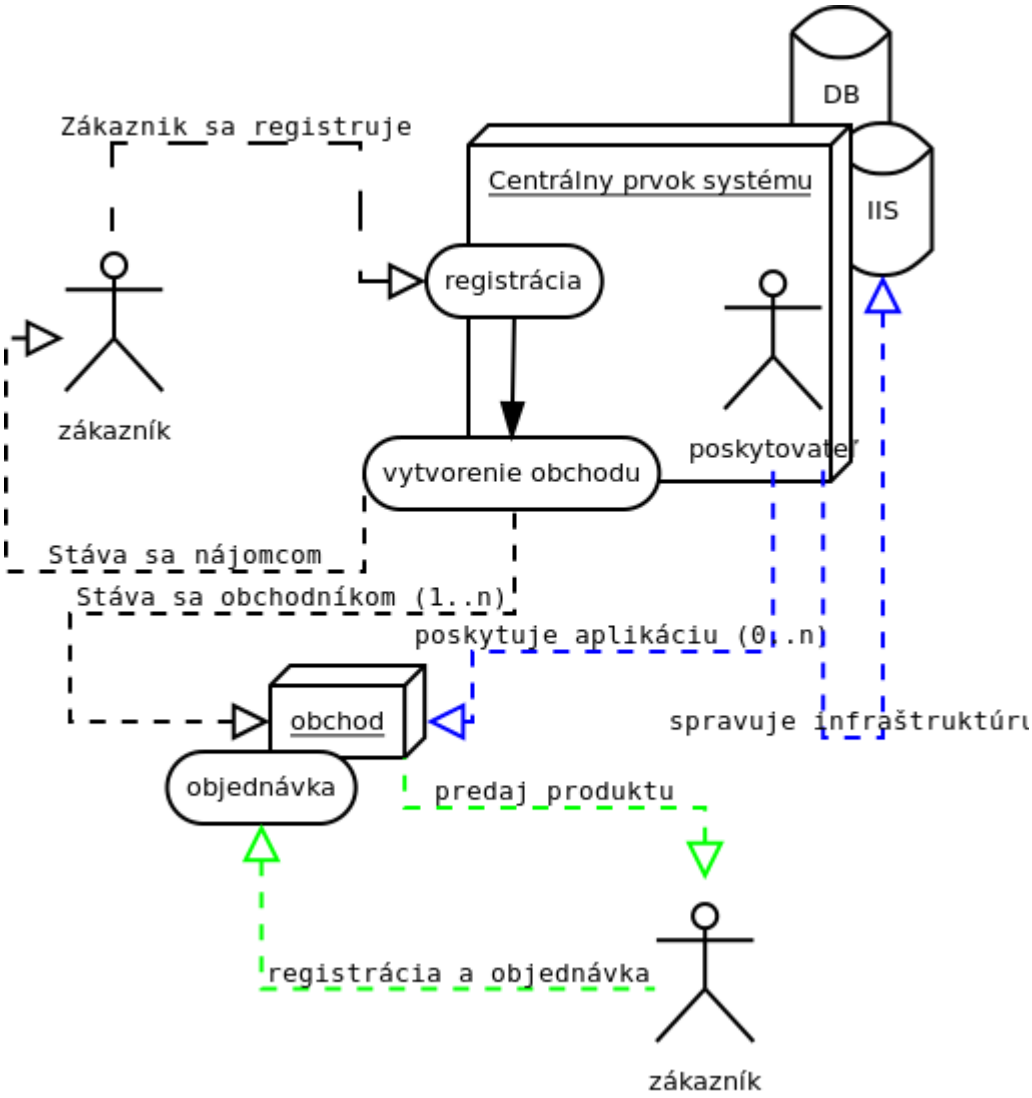
Špecifickejšie centrálny prvok v tomto konkrétnom riešení ponúka poskytovateľovi prehľad vytvorených obchodov, zoznam registrovaných užívateľov alebo zmenu prihlasovacieho hesla. Vhodnými prvkami správy celého systému je aj zablokovanie prístupu užívateľovi, či chodu elektronického obchodu z administrátorského prostredia po prihlásení a autorizácii.

Registrácia – vytvorenie užívateľského účtu spolu so založením obchodu, teda vytvorenie odberateľa služby, je dostupná priamo na domovskej stránke. Po autentifikácii a autorizácii sú mu sprístupnené voľby editácie osobných informácii, zmeny hesla, a aj založenie ďalšieho obchodu s unikátnym menom v rámci systému. Nemenej využívanou stránkou pre prenajímateľa obchodov je ich zoznam s odkazmi na administratívne rozhranie obchodu a prezentáciu obchodu.

Správa obchodu prebieha po dodatočnom prihlásení. Oproti bežným funkciám ako je typ nastavenia, správa kategórii, zákazníkov, produktov, objednávok, ponúkne tvorbu vzorov produktov, rozšírenú logistiku zahrňujúcu užívateľom vytvárané spôsoby platby, dodania a ich asociácie. Administráciu zamestnancov, s možnosťou zaradenia do rolí. Posledným celkom správy je na implementáciu najnáročnejšia časť: editor vzhľadu prezentácie, obsahujúci tvorbu i zdieľanie vzorov vzhľadu a rozložení prvkov.

Prezentačná časť je určená na zobrazovanie kategorizovaných produktov, alebo ich detail. Zabezpečuje digitálny košík a pokladňu pre vytvorenie objednávky. Tiež zákazníkom po prihlásení alebo registrácii sprístupní ich zoznam nákupov (resp. stav objednávok).

Vyššie spomenuté funkcie zabezpečujú dostatočnú robustnosť pre distribuovaný obchod a nutné minimum pre centrálny prvok. Toto nastavenie vzniklo na predpoklade rušnej komunikácie hlavne na jednotlivých obchodoch a občasných zmenách vo vrstve medzi poskytovateľom a odberateľom služby. Nadštandardné funkcie správy obchodu majú za cieľ rozšíriť pokrytie malých, stredných obchodov ako potenciálnych užívateľov služby pre začínajúcich poskytovateľov softvéru ako služby využívajúcich Generátor elektronických obchodov. ....(vytýčenie vlastností centrálneho prvku a obchodov)



(Obrázok 5 – biznis model schéma – poskytovateľ – klient – klient)

1.4. Existujúce riešenia

- **Vltava2000** je udomácnená už dlhé roky na českom e-commerce trhu. Najprv jej produktom bol krabicový elektronický obchod. Jeho administrácia prebiehala cez desktopovú aplikáciu. No v nedávnom období, svoju aplikáciu začali distribuovať ako službu. Obsahuje základné prvky elektronického obchodu. Na zmenu vzhľadu ponúkajú výber z niekoľkých šablón. Služba je spoplatnená a jej mesačná cena závisí od zvolenej konfigurácie.
- **WebNode** tento projekt začal najprv ako CMS ponúkané ako služba. No s jeho ďalším rozvojom pribudli aj funkcie elektronického obchodu. Odberateľ má možnosť upraviť vzhľad obchodu výberom z mnohých šablón. Následne v bohatom editore vzhľadu a obsahu mení usporiadanie modulov a vzhľad niektorých grafických prvkov. Služba je v základnej verzii poskytovaná zdarma. Toto kvalitné riešenie pochádza z Čiech a je úspešné aj v zahraničí.
- **Tiger Commerce** veľmi atraktívne riešenie. Ponúka administrátorské rozhranie s mnohými unikátnymi užívateľskými prvkami. Cenovo vychádza

obdobne ako Vltava2000. Žiaľ pre český trh chýba lokalizácia do českého jazyka.

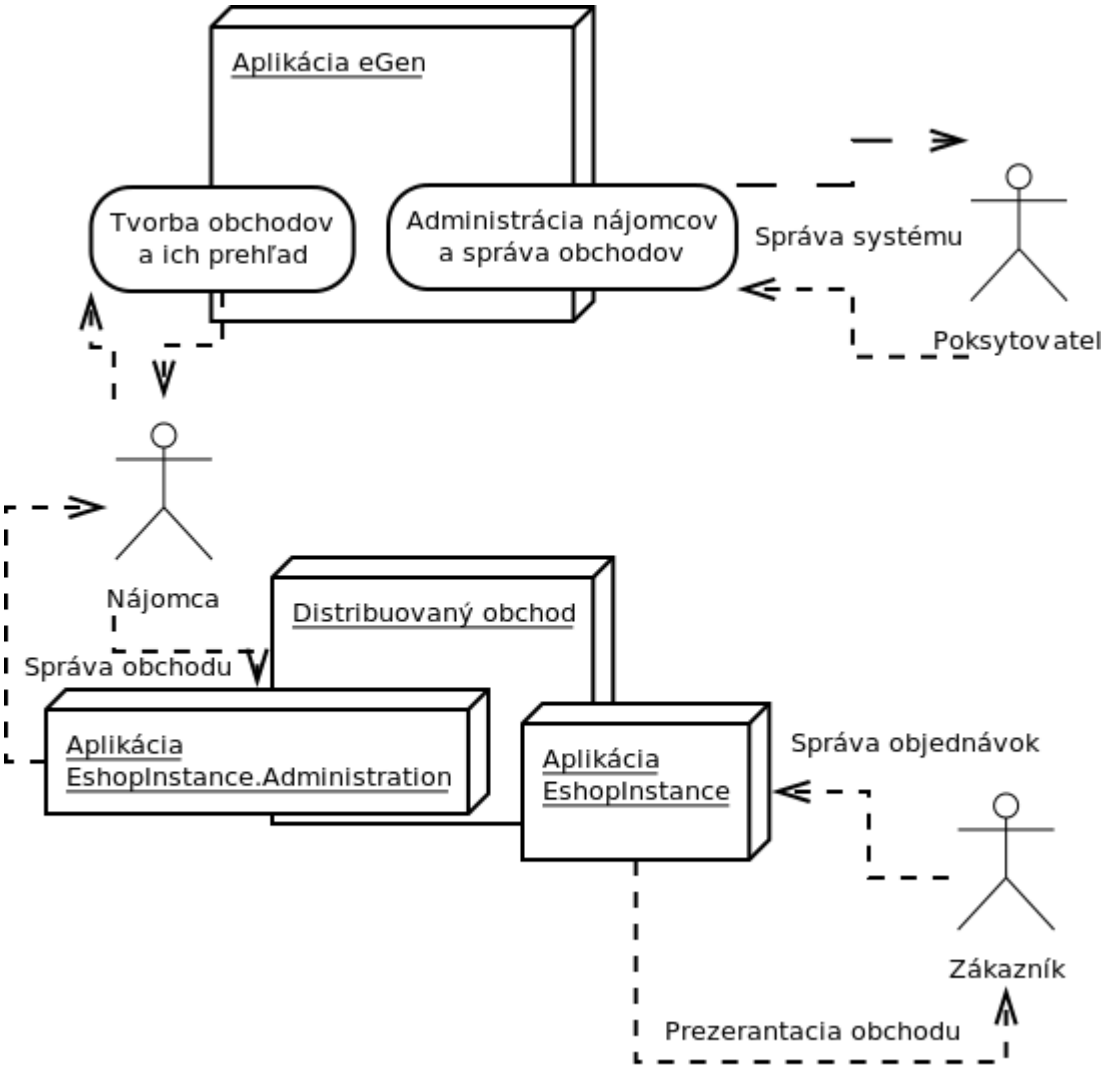
2. Užívateľská dokumentácia

V tejto časti sa zoznámime bližšie s aplikačnou vrstvou a popíšeme prácu s grafickým rozhraním. To je vytvorené tak, aby bolo dostupné cez novšie webové prehliadače ako Firefox 3.6, Chrome (12.0), ktoré sú dostupné zdarma. Oboznámime sa s postupmi vytvárania obchodu a jeho správy. Ukážeme si možnosti pri tvorbe vzhľadu. Popri tom si sem tam uvedieme iné alternatívy k práve rozoberaným riešeniam.

2.1. Celky užívateľského rozhrania

Užívateľské rozhranie kopíruje biznis model samotného riešenia. Komunikačným rozhraním centrálneho prvku sú z pohľadu užívateľa webové stránky aplikácie eGen. So spomenutými stránkami pracuje poskytovateľ služby aj odberatelia. Ich nosnou úlohou je vytvárať jednotlivé distribúcie obchodov s vlastným rozhraním, preto túto časť označíme menom „Generátor“. Popri tejto úlohe zabezpečuje prístup k zdieľaných dátam medzi aplikáciami pomocou webovej služby. Vďaka tomu majú možnosť využívať jednotlivé distribúcie obchodov medzi sebou vzory vzhľadu a rozloženia prvkov.

Druhá časť modelu, ktorá rieši obchod a jeho zákazníkov je pokrývaná dvomi navonok samostatnými užívateľskými rozhraniami pracujúcimi nad konkrétnou distribúciou. Prvé zabezpečuje správu obchodu. Pomenovali sme ho „Administrácia obchodu“. Jeho nosným prvkom je aplikácia EshopInstance.Adminstrator. Druhá časť prezentujúca produkty obchodu zákazníkom nesie príznačné meno „Prezentácia obchodu“. Dušou tohto celku je softvér menom EshopInstance. Rozdelenie nám ilustruje Obrázok č.6.



(Obrázok č.6 – rozdelenie užívateľských celkov nad biznis modelom)

Toto rozdelenie je prirodzené vzhľadom na zvolenú architektúru. Ponúka zaužívané pohľady v elektronických obchodoch s minimálnou prepojenosťou na Generátor. Čím nájomcovi umožňuje slobodnejšie budovať jeho obchody.

V ranných náčrtoch bola uvažovaná myšlienka jedného spoločného užívateľského rozhrania pre registrovaných užívateľov celého systému. Teda jedno rozhranie ponúkajúce funkcie na základe autorizácie užívateľa. Jedna z veľkých výhod bola pohodlnosť koncového zákazníka obchodu, ktorý by mal na jednom mieste prístupné objednávky z rôznych obchodov. Dokonca by mohol byť pod jedným účtom aj prenajímateľom obchodu. Žiaľ, napriek peknej predstave, prevládali obavy z nedostatočnej rozšírenosti tohto prístupu. Z toho plynúcej zmätenosti užívateľov, ktorý by boli nútení všímať si pri registráciách do obchodov, či nie sú v danej rodine registrovaní. Tiež by to budilo dojem obchodníka ako súčasť celku pod záštitou poskytovateľovej spoločnosti, čím by prenajímateľ stratil kúsok svojej identity.

Za pozornosť by stálo hybridné riešenie so snahou vyťažiť maximum výhod z oboch riešení.

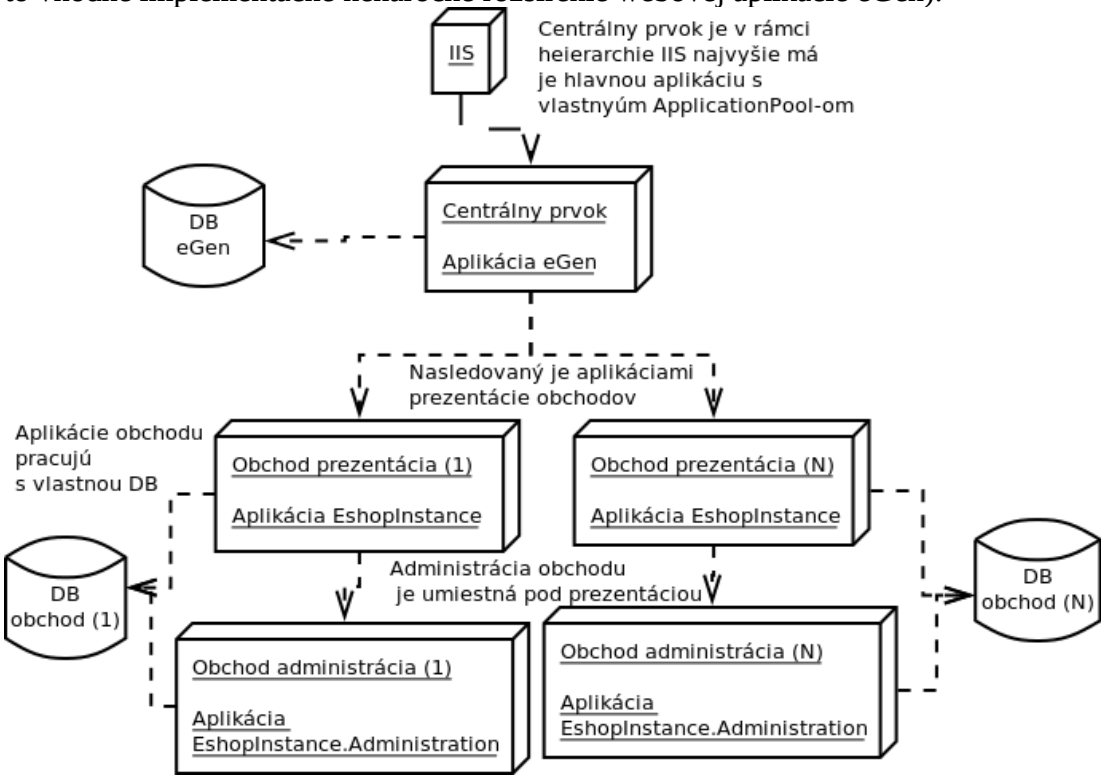
2.2. Architektúra systému

Hlavná aplikácia eGen interpretovaná cez IIS 7.0. má vlastnú databázu v MS SQL serveri a celkový vzhľad architektúry sa odvíja od toho, akým spôsobom vytvára a poskytuje softvér ako službu.

Pri tvorbe novej distribúcie obchodu sa vytvorí nová aplikácia so zadaným menom obchodu v IIS hierarchicky umiestnená pod aplikáciou centrálneho prvku. No ešte predtým sa vytvorí nová databáza. Naplní sa inicializačnými dátami nutnými pre chod ostatných aplikácii tvoriacich toto riešenie. K týmto dátam popri iných patrí skopírovanie prihlasovacieho mena a hesla užívateľa vytvárajúceho obchod. Takýto užívateľ bude mať prístup do celej správy obchodu. Následne sa fyzicky v adresári na tom určenom nainštalujú aplikácie EshopInstance a EshopInstance.Administration. Teda vytvorí podľa vzoru vlastnú adresárovú štruktúru, skopíruje príslušné súbory a upraví ich konfiguráciu. Po týchto operáciách generátor spustí aplikáciu v rámci IIS, čím bude služba plne dostupná prenajímateľovi.

Z pohľadu architektúry celý systém tvorí jedna centrálna webová aplikácia eGen a mnoho webových aplikácii (tvoria prezentačnú vrstvu) s vlastnou adresárovou štruktúrou a databázou. Sú závislé na knižnici EshopInstance.Data poskytujúcej dátové modely a prístup do databázy. EshopInstnace.Repository tvorí vrstvu s dotazmi nad dátovou vrstvou a EshopInstance.Services implementuje biznis logiku aplikácie. V prípade dotazu na obchod IIS zabezpečí, aby ho riešila práve jemu patriaca webová aplikácia.

Týmto spôsobom bola dosiahnutá nezávislosť jednotlivých obchodov. Nefunkčnosť jedného neovplyvní beh ostatných. V prípade, ak by nájomca potreboval špeciálnu starostlivosť od poskytovateľa, napríklad prechod na samostatný server alebo dodatočné úpravy funkčnosti obchodu na mieru klienta, tak vďaka tomuto návrhu s tým nebude mať problém. Veľkou výhodou je možnosť aktualizovať obchody jednotlivo, teda poskytovať aktualizácie klientom podľa ich vzťahu k poskytovateľovi (moje riešenie neposkytuje automatické aktualizácie, no je to vhodné implementačne nenáročné rozšírenie webovej aplikácie eGen).



(Obrázok č. 7 – architektúra systému)

V opozícii tohto návrhu je alternatíva vybudovať jednu mohutnú webovú aplikáciu s jednou databázou a adresárovou štruktúrou. Identifikácia obchodov by neprebiehala na úrovni IIS ako v použitom modeli, ale na úrovni aplikácie pomocou parametru v URL špecifického pre každý obchod. Tiež generovanie obchodu by bolo čisto doménou vloženia nového riadku do databázy, čo by výrazne zjednodušilo tento proces. Nutnosť vytvoriť pre obchod unikátnu adresárovú štruktúru by zostala, alebo by bola simulovaná prefixami súborov (veľa súborov v jednom adresári znamená pomalší prístup k nim), keďže jednotlivé obchody by mali vlastné súbory s kaskádovými štýlmi, obrázkami a inými zdrojmi tvoriacimi identitu obchodu (za predpokladu, že by neboli generované alebo umiestnené v DB). Možnosti poskytovateľa ponúkať špeciálne služby prípadným rýchlejšie sa rozvíjajúcim nájomcom by boli limitované prepojením generátora a obchodu. V prípade chyby by táto architektúra znefunkčnila poskytovanie služby všetkým odberateľom [13].

2.3. Generátor

Generátor je v zastúpení webovej aplikácie eGen postavenej na technológii ASP.NET. Naprogramovaná je v jazyku C#. Má trojvrstvovú architektúru, ktorú som zvolil najmä pre vyššiu prehľadnosť kódu aplikácie. Najnižšou vrstvou je dátová vrstva, obsahujúca dátové modely a dotazy na databázu. Dátové modely aj samotnú komunikáciu zabezpečuje objekt triedy DataSet z rodiny ADO.NET, ktorý pomocou zabudovaných nástrojov Visual Studia (nástroj použitý pri implementácii) automaticky vygeneruje triedy DataTable reprezentujúce tabuľky databázy aplikácie a tiež zabezpečí triedy typu DataAdapter, ktoré slúžia na dotazovanie databázy [14].

Stredná vrstva tvorí biznis logiku aplikácie. Je to akýsi pomyselný most medzi prezentačnou a dátovou vrstvou, ktorý môže prepravovaný náklad z jednej strany na druhú vrátiť alebo transformovať pre potreby druhej strany. Tu sa napríklad odvíja spôsob, akým overujeme prístup a oprávnenia užívateľom, alebo procesy vedúce k vytvoreniu novej distribúcie obchodu.

Účelom poslednej vrstvy je interakcia s užívateľmi aplikácie. Prezentačná vrstva vytvára dynamické html stránky, získava užívateľove vstupy a poskytuje webové služby.

Výsledkom tohto spojenia sú nasledujúce hlavné funkcie aplikácie eGen:

- Možnosť registrovať prenajímateľa a vytvoriť elektronický obchod.
- Pomocou administrátorského účtu zablokovať a odblokovať zákazníkovi prístup k účtu alebo samotný obchod.
- Pohľady na dáta zákazníkov a ich obchody.
- Poskytovanie webovej služby „WS_WebPageTemplate.asmx“ na zdieľanie a prácu so vzormi vzhľadu a rozloženia prvkov obchodu.

Bezpečnosť pred neoprávneným užívaním rieši autentifikácia pomocou prihlasovacieho mena a hesla. Následne poskytuje funkcie podľa oprávnení užívateľa (autorizácia). Pracuje s dvomi zabudovanými rolami:

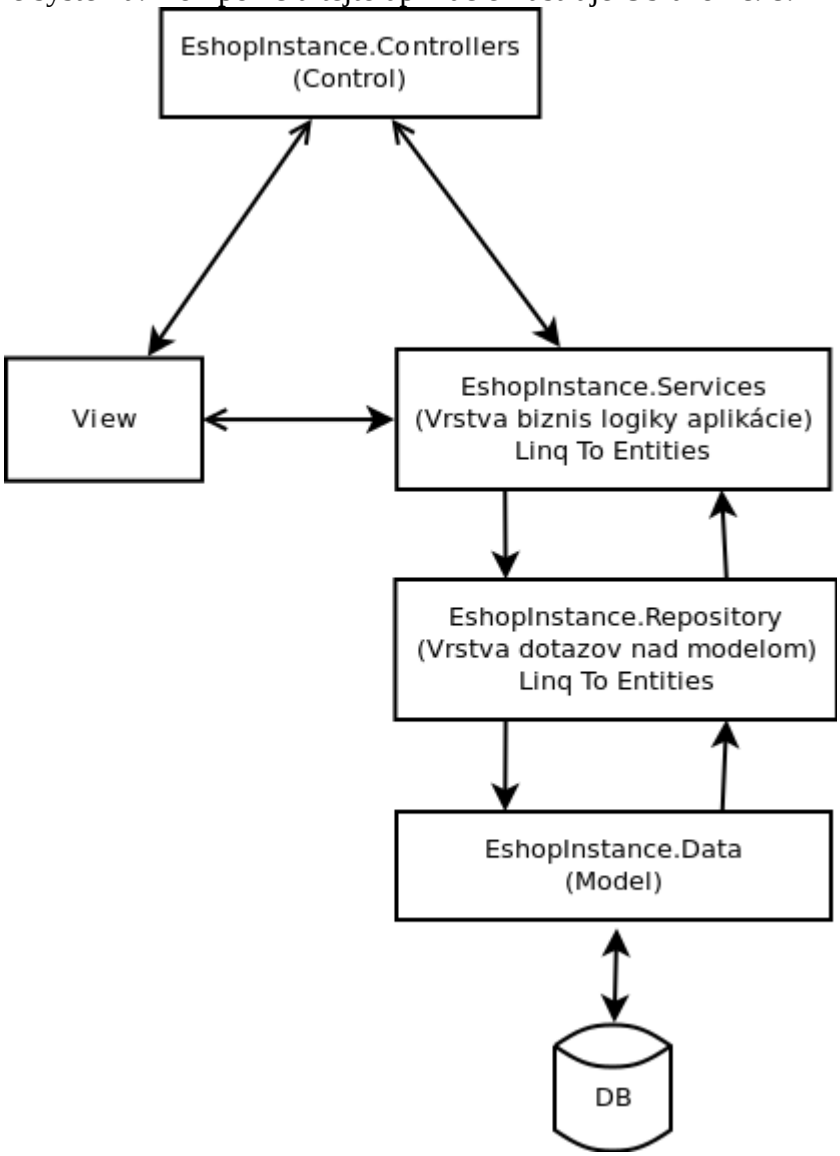
- Administrátor – rola patriaca užívateľovi povereného od poskytovateľa na správu obchodov a ich nájomcov. Tento užívateľ je vytvorený po inštalácii

systému. Má prihlasovacie meno „root“ a heslo „sample“, ktoré je vhodné zmeniť.

- Zákazník – táto rola patrí registrovaným užívateľom, nájomcom obchodov.

2.4. Administrácia obchodu

Aplikácia EshopInstance.Administration sa odlišuje od vyššie spomenutej nielen svojou architektúrou, ktorá je postavená na modely MVC (Model – View – Control) v spojení viacvrstvového návrhového vzoru [16], ale aj použitými technológiami ako Entity Framework [17] alebo ASP.NET MVC Framework[18]. Napísaná je rovnako v jazyku C# pre behové prostredie .NET 3.5. Jednotlivé nižšie vrstvy tvoria samostatné celky reprezentované dynamickými knižnicami, ktoré boli spomenuté pri architektúre systému. Kompozíciu tejto aplikácie ilustruje Obrázok č. 8.



(Obrázok č. 8 – architektúra aplikácie, administrácie a prezentácie obchodu)

Zvolené riešenie pomocou ASP.NET MVC Frameworku predstavovalo väčšiu implementačnú voľnosť oproti klasickému ASP.NET. Urýchlilo integráciu a

prácu s AJAX a javascript-ovými knižnicami ako jQuery. Najrozsiahlejšia časť „editácia vzhľadu“ by principiálne bola riešená rovnako, no v prípade ASP.NET by si vyžadovala viac času na naprogramovanie.

Tvorba vzhľadu prebieha v editore, ktorého výsledkom je súbor s kaskádovými štýlmi určujúci vzhľad obchodu a štruktúra popisujúca rozloženie prvkov pre jednotlivé stránky prezentácie. Prvkami stránky rozumieme bloky a moduly. Moduly tvoria logické celky s určitou funkciou (Napr. funkcia rýchleho prihlásenia je zabezpečená modulom Rýchle prihlásenie). Bloky sú kontajnery, do ktorých sa moduly umiestňujú. Viac o editore je popísané v samostatnej časti 3.6. Editácia vzhľadu.

V ponuke zaujímavých prístupov aplikácia pokračuje vytváraním vzorov produktov, ktorých úloha je jednoducho parametrizovať produkty rovnakého zamerania (Napríklad pre produkty notebookov sa vytvorí vzor notebook s parametrami ako procesor, hdd a iné). Detailnejšie je to popísane v 3.7. Tvorba produktových vzorov.

Nezanedbateľnou časťou správy je aj logistika obchodu. Aj keď sa táto časť nemení často, v prípade distribuovaného obchodu musí ponúkať dostatočne flexibilné nástroje pre pokrytie širokej ponuky v tejto oblasti. Preto súčasťou aplikácie je možnosť vytvárať sklady, obchody, či ľubovolné spôsoby doručenia a platby. Následne vytvárať vzťahy medzi nimi, ktoré určujú, pre ktoré spôsoby doručenia sú prístupné spôsoby platby.

Bezpečnosť je riešená obdobne ako pri aplikácii eGen. Do správy obchodu má najprv prístup len užívateľ v role „vlastníka“ - prenajímateľ obchodu, ktorého prihlasovacie meno a heslo bolo skopírované pri tvorbe obchodu z aplikácie eGen. Jeho autorita mu umožňuje prístup ku všetkým funkciám aplikácie. Na vytváranie iných užívateľov je určená sekcia Správa zamestnancov. Tá umožňuje vytvárať užívateľov v roly „Zamestnanec“ a prideliť im dodatočné role podľa ich pôsobiska v rámci správy obchodu. Rozoznávame tieto role:

- „designer“ - dáva oprávnenie na prístup a prácu s funkciami sekcie Web, kde sa nachádza editor vzhľadu, vzory vzhľadu a iné stránky.
- „personalist“ - už podľa názvu možno usudzovať, že táto rola otvára cestu k nástrojom Správy užívateľov (Zamestnanec s prístupom do správy zamestnancov môže vytvoriť zamestnancov len s oprávneniami, s ktorými sám disponuje).
- Rola „logistic“ sprístupňuje nástroje logistiky obchodu.
- Pre držiteľov oprávnenia „product“ sú k dispozícii sekcie Vzory (produktové vzory) a Produkty
- Poslednou rolou je „order“, ktorá ponúka pohľad a funkcie sekcie Objednávky.
- Popri spomenutých možnostiach aplikácia ponúka:
- Dynamickú tvorbu a editáciu kategórii v strome.
- Správu produktov so zaradením produktu do kategórie, vzoru a jednotlivých skladov a obchodov.
- Automatické pridávanie výrobcov produktu, pri jeho tvorbe.
- Základné nastavenia obchodu.
- Správu objednávok v podobe zobrazenia detailu s objednaným tovarom a možnosťou meniť jej stav.
- Jednoduchý prehľad zákazníkov obchodu.

- Pohľad na dostupné vzory vzhľadu a ich použitie.
- Jednoduché verziovanie pri úprave vzhľadu.
- Kopírovanie štýlov pri úprave vzhľadu schémy medzi blokmi alebo stránkami.

2.5. Prezentácia obchodu

Aplikácia EshopInstance je prezentáciou obchodu a komunikuje s jeho zákazníkmi. Avšak k tejto bežnej činnosti elektronických obchodov musí pristupovať odlišne. Ako sme vyššie uviedli, dôležitým prvkom obchodu je budovať vlastnú identitu. To čo sme vytvorili v editore je nutné aplikovať v tejto prezentácii a vygenerovať odpovedajúcu štruktúru stránky podľa dát získaných zo spoločnej databázy obchodu.

Popri tejto pre užívateľa nepostrehnuteľnej aktivite sa aplikácia stará o registráciu zákazníkov v role „zákazník“. Tí majú prístup k vyhotoveniu objednávky, ktoré prebieha v troch krokoch za predpokladu neprázdneho košíka. Prvým je výber spôsobu platby a dodania, pomocou AJAX vyplnené údaje uloží na severi, aby sa k tomuto kroku mohol užívateľ prípadne vrátiť. Nasleduje krok s fakturačnými údajmi a potvrdenie objednávky.

Táto aplikácia používajúca rovnaké technológie ako EshopInstance.Administration ponúka:

- Pohľady na objednávky zákazníka.
- Základnú správu užívateľského účtu ako zmenu hesla a osobných údajov.
- Systémové stránky s obchodnými podmienkami obchodu, postupom pri reklamácii a stránku kontaktov obchodu vypisujúcu sklady a obchody s uvedenými adresami.

2.6. Knižnica EshopInstance.Services

Slúži ako podpora pre vyššie spomenuté MVC aplikácie. Ide o dynamickú knižnicu napísanú v jazyku C#. Obsahuje rôzne triedy. Slúži na získanie modelov, validáciu dát a iných funkcií týkajúcich sa logiky aplikácie.

Jednotlivé triedy sú rozdelené na „Models“, „Utility“, „WebGui“ a ostatné. „Models“ tvoria generické typy. Používané napríklad pre stránkovanie modelov v prezentačnej vrstve. Triedy ako ErrorLog (poskytujúca záznam chýb) alebo Configer, cez ktorý je vhodné pristupovať ku konfiguračným súborom aplikácie sa nachádzajú v mennom priestore „Utility“. Zaujímavejší je menný priestor „WebGui“ s triedami, ktoré pokrývajú nástroje nutné na editáciu vzhľadu a zmenu štruktúry prezentácie. Medzi ostatnými triedami nájdeme triedu starajúcu sa o užívateľov, produktov a iných modelov, ktoré možno ovplyvňovať užívateľmi obchodu na úrovni prezentačnej vrstvy.

2.7. Knižnica EshopInstance.Repositories

S touto vrstvou pracuje najmä vrstva služieb. Obsahuje dotazy nad jednotlivými modelmi Enitity Frameworku. Tiež ide o dynamickú knižnicu napísanú v jazyku C#. Je vítané aby používali jednu inštanciu Adaptéra databázového modelu Entity

Frameworku, lebo v prípade dvoch dotazovaných modelov nad rôznymi inštanciami nebudeme môcť vytvárať medzi nimi prípadné asociácie. Toto obmedzenie vychádza z kontextu Entity Frameworku, kde si každá inštancia Adaptéru vedie vlastné záznamy o zmenách jednotlivých modeloch. Tu by však nastal problém v prípade, že by mu bol dodaný objekt o ktorom nemá žiadny záznam. Preto objekty knižnice služieb a vrstvy dotazov nad úložiskom („repository“) sú vo väčšine prípadov konštruované s parametrom objektu adaptéru databázového modelu (ten je vytvorený vo vrchných častiach aplikácie).

2.8. Knižnica EshopInstance.DataAccess

Ide o dynamickú knižnicu obsahujúcu triedy reprezentujúce prvky databázy obchodu. Tie boli vygenerované Visual Studiom pre potreby Entity Frameworku. Takýmto spôsobom vznikli dva Adaptéry databázového modelu obchodu. Prvý, ktorý sa zaoberá tabuľkami bežného riadenia obchodu, je EshopInstanceDBModel. Druhý je WebAppereanceModel, ktorý pokrýva tabuľky určujúce vzhľad a štruktúru aplikácie.

2.9. Inštalácia, požiadavky

Samotná inštalácia systému na infraštruktúre poskytovateľa je popísaná krok za krokom v Prílohe A. No požiadavky radšej znovu stručne uvediem. Nutný je operačný systém Windows Server 2008, alebo Windows 7 s nainštalovaným IIS vo verzii 7.0. Presnejšie funkcie, ktoré musí mať web server spustené sú uvedené v Prílohe A. Ďalšou nutnou súčasťou je .Net Framework 3.5.1 a ASP.NET MVC Framework 1.0. Pre ukladanie dát bol zvolený databázový server MS SQL 2008 Express Edition. Aplikácia si vyžaduje právo čítať a zapisovať do súborov redirection.config a applicationHost.config. Rovnaké práva si vyžadujú aj adresáre, do ktorých bude generátor vytvárať obchody, alebo serializovať objekty reprezentujúce vzory vzhľadu a štruktúry.

2.10. Základné použitie Generátora

Hlavná stránka prezentuje registračný formulár pre nového odberateľa služby obsahujúci nutné polia pre vytvorenie nového obchodu a užívateľa – teda meno obchodu prijateľné pre použitie v URL. Potom prihlasovacie meno a heslo. Po odoslaní formulára sa jednotlivé vstupy validujú a v prípade správneho vyplnenia prihlási užívateľa, a presmeruje ho na príslušnú stránku podľa jeho role. V prípade chýb vo formuláre sa zobrazia na stránke nápovedy pri jednotlivých chybných vstupoch.

Pre orientáciu v aplikácii slúži navigácia obsahujúca odkazy na ďalšie stránky aplikácie. Medzi stránky dostupné pre všetkých patrí stránka s prihlasovacím formulárom. Tú využívajú na prihlásenie rovnako poskytovateľ služby – administrátor Generátora a aj nájomca.

Správcovi úspešná autorizácia prinesie prístup k stránke zoznamu vytvorených obchodov („Obchody“) a stránke nájomníkov („Nájomníci“). Zoznam

obchodov obsahuje v riadu preklik na zobrazenie detailu obchodu. Tento detail obsahuje okrem informácii odkaz na nájomcu a tlačítko „zablokovať“ resp. v prípade zablokovaného obchodu tlačítko „odblokovať“. Účel zrejme netreba popisovať.

Obdobné prvky sa nachádzajú aj v zozname nájomcov. V ich detaile je okrem informácii aj zoznam ich obchodov a tlačítko na zablokovanie alebo odblokovanie prístupu užívateľa.

Posledným no o to dôležitejším odkazom v navigácii administrátora je „zmena hesla“. Stránka ponúka tri textové vstupy. Jeden na pôvodné heslo a zvyšné na nové a jeho overenie.

Druhou skupinou odkazov navigácie zobrazovaných len nájomcom sú odkazy:

- „Údaje“ so stránkou na editáciu osobných údajov pomocou formulára a jeho odoslania.
- „Zmena hesla“ rovnaká ako pri administrátorovi (treba si uvedomiť, že so zmenou hesla nedôjde k zmene hesiel pre prihlásenie nájomcu do jednotlivých obchodov).
- „Zoznam mojich obchodov“ stránka obsahujúca nájomníkove obchody a odkazy na ich prezentačnú a administratívnu webovú aplikáciu.
- „Vytvoriť obchod“

Najdôležitejšou funkciou Generátora je samotné vytvorenie obchodu. To prebieha buď pri registrácii užívateľa alebo na stránke „Vytvoriť obchod“ obsahujúcej jednoduchý formulár na zadanie mena obchodu, kratšieho ako 50 znakov, zloženého z abecedných písmen, čísel, alebo pomlčky, no nie umiestnenej na začiatku názvu.

2.11. Základné použitie administrácia obchodu

Aplikácia prináša rozsiahle užívateľské prostredie. S tým sú spojené aj rozsiahle nástroje a možnosti ich použitia. Rozdelené sú logicky podľa ich funkcie do rôznych sekcií, ktoré v podstate kopírujú role používané v aplikácii.

Aby sme mohli pracovať s aplikáciou je nutné zadať prihlasovacie meno a heslo. V prípade úspešnej autentifikácii sa nám zobrazí jednoduchá navigácia, kde každý odkaz je bránou do jednotlivých sekcií.

2.11.1. Správa zamestnancov

Prvou sekciou je správa zamestnancov. Jej úvodná stránka nám ponúka pohľad na zamestnancov a odkaz na pridanie zamestnanca. Odkazom „Pridať nového“ sa dostaneme na stránku s formulárom pre pridanie zamestnanca. Okrem bežných textových vstupov môže obsahovať zaškrtávacie políčka s nápismi:

- Sekcia Web - editácia vzhľadu
- Správa logistiky
- Správa objednávok
- Správa zamestnancov

Daný zoznam je závislý od užívateľovi prístupných sekcií (resp. zaradením do rolí). Takýmto spôsobom sme zabezpečili možnosť pridávať nových

zamestnancov len do sekcií, na ktoré má pridávajúci užívateľ oprávnenie. Opísaný prístup ponúka prenajímateľovi možnosť vytvárať v rámci obchodu hierarchiu medzi zamestnancami. Aj keď hlavne v prípade role „personalist“, ktorý pokrýva práve sekciu správy zamestnancov možno namietať, že niekto, kto sa má starať len o prístupy zamestnancov, je nútený byť členom aj ostatných sekcií, aby mohol ľubovoľne pracovať so zaradením zamestnancov. Po správnom vyplnení textových polí a zaškrtnutých sekcií, ktoré chceme novému zamestnancovi prideliť, nás aplikácia presmeruje na hlavnú stránku sekcie.

Vytvorený zamestnanec nemá povolený prístup a má heslo „sample“, ktoré je vhodné si po jeho prvom prihlásení zmeniť. Aplikácia každého užívateľa so základným „sample“ heslom po prihlásení vyzýva na jeho zmenu. Uvedené správanie aplikácie nie je bezpečné a odporúčam ho prípadným užívateľom môjho riešenia zmeniť. Prvotný návrh rátal s bezpečnejším modelom v podobe generovania hesla a následného odoslania na e-mail nového zamestnanca. Väčšia časť takéhoto riešenia je implementovaná, no z dôvodu nedostatočnej infraštruktúry pri vývoji, na ktorej som nemal možnosť overiť danú funkčnosť, som zvolil jednoduchšiu variantu.

Hlavná stránke obsahuje prehľad zamestnancov. Ten má podobu tabuľky so stĺpcami s informáciami o zamestnancoch. Dôležitým stĺpcom je „Stav prístupu“. Jeho obsah pri jednotlivých užívateľoch môže nadobúdať tri hodnoty:

- „Povolený“ ak má užívateľ povolený prístup. Hodnota v bunke tabuľke je v tomto prípade aktívna. Kliknutím na ňu sa asynchrónnym volaním na server pomocou jQuery (AJAX) zmení užívateľovi stav prístupu na „Zakázaný“. Túto operáciu sprevádza informačné dialógové okno.
- „Zakázaný“ je prípad, keď má užívateľ zakázaný prístup do aplikácie. Jeho autentifikácia bude neúspešná. Systém ho v prípade pokusu o prihlásenie oboznámi s jeho situáciou. Aj táto hodnota je aktívna. Kliknutím na bunku nastavíme stav prístupu na „Povolený“. Akcia je obdobná ako v predchádzajúcej situácii.
- „--“ zobrazí v prípade prenajímateľa obchodu (v rámci obchodu ho často pomenovávam aj majiteľom). Na rozdiel od predchádzajúcich prípadov nie je bunka tabuľky tejto hodnoty aktívna. Prístup prenajímateľovi nie je možné zakázať. Je to ošetrené na úrovni vrstvy služieb v rámci triedy EmployeeService, ktorá disponuje metódami reflektujúcimi funkčnosť užívateľského rozhrania.

Posledným nástrojom v tejto sekcii je možnosť editovať údaje zamestnanca. Ponúka obdobný formulár ako pri pridaní užívateľa až na možnosť zmeniť prihlasovacie meno. V prípade úpravy údajov majiteľa obchodu nie je možné meniť jeho zaradenie do sekcií.

2.11.2. Logistika

Táto sekcia je bohatá na klasické formuláre s textovými vstupmi. Obsahuje niekoľko podsekcií:

- „Prevzatie a platby“
- „Dodávatelia“
- „Obchody a sklady“

„Dodávatelia“ a „Obchody a sklady“ poskytujú náhľady na ich dáta a možnosť ich

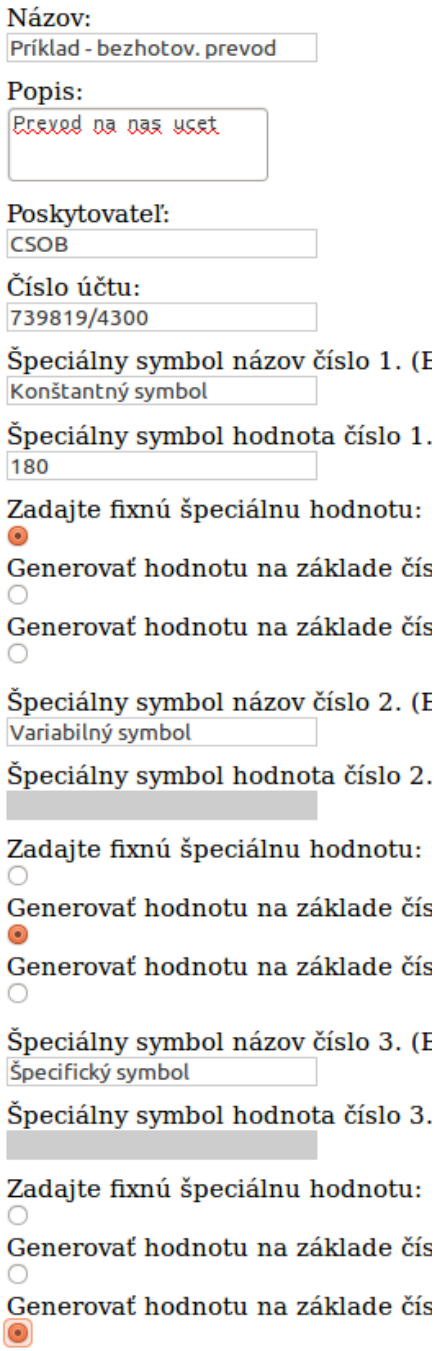
tvorby a úpravy, pričom aplikácia v prípade zlých vstupov užívateľa usmerní. Záznamy o dodávateľoch sú určené na rozšírenie informácií o produkte a pre zákazníkov sú nedostupné. Tvoria vnútornú agendu obchodu. Obdobne sú použité sklady a obchody.

Podsekcia „Prevzatie a platby“, ako jej názov napovedá, obsahuje dáta určené pre nutné kroky zákazníka v pokladni obchodu. Stránka obsahuje odkaz na stránku s spôsobu formulárom pridania prevzatia. Ďalší odkaz nás nasmeruje na stránku pridania spôsobu platby. Tu popri povinných vstupoch „Názvu“ a „Popisu“ máme možnosť vytvoriť spôsoby platby, ktoré dynamicky vytvárajú záverečné inštrukcie pre zaplatenie objednávky v pokladni zákazníkom. Ako na to? Popíšeme si príklad, ktorý nám objasní takéto použitie.

Príklad: chceme mať spôsob zaplatenia bezhotovostným prevodom na účet „739819/4300“ s konštantným symbolom 180, variabilným symbolom s číslom objednávky a špeciálnym symbolom obsahujúcim identifikátor užívateľa. Riešením je vyplniť formulár ako na Obrázku č. 9.

Dôležité sú najmä položky začínajúce textom „Špeciálny symbol názov“ a „Špeciálny symbol hodnota“. Prvé definujú názov a druhé hodnotu. Máme možnosť vytvoriť tri takéto páry, pričom pre políčko hodnoty máme v ponuke zadať hodnotu napevno, alebo vybrať si dynamickú hodnotu, ktorá bude zobrazená užívateľovi pri dokončení objednávky. Dynamické hodnoty sú buď číslo objednávky alebo identifikačné číslo užívateľa.

Keď máme vytvorené spôsoby prevzatia aj platby, je nutné vytvoriť medzi nimi vzťah, pričom spôsobu prevzatia prideľujeme 1 až n spôsobov platieb. Spôsoby prevzatia sú zobrazené v logistickom kroku pri vytváraní objednávky v prezentačnej časti obchodu, iba ak majú aspoň jednu asociáciu so spôsobom platieb (Potom je aj spôsob platby dostupný v prezentácii obchodu). Ak nie je vytvorený žiadny asociovaný spôsob prevzatia a platby, tak je nefunkčná tvorba objednávky v prezentácii obchodu.



(Obrázok č. 9 – spôsob platby – prevodom)

2.11.3. Správa kategórii

Sekciu nájdeme pod odkazom navigácie „Kategórie“. Účel kategórií v elektronických obchodoch každý dobre pozná. Ich použitie v systéme je intuitívne. Vykonáva sa pomocou úpravy stromu kategórii. Základnou funkciou je pridanie potomka kategórie (systém pracuje s koreňovou kategóriou, ktorá je pre užívateľov nedotknuteľná). Kliknutím na „Pridať kategóriu“ sa nám otvorí dialógové okno so vstupom pre jej meno. Po jeho zadaní odobríme pridanie nového potomka kategórie na rovnakú úroveň v akej sa nachádza voľba „Pridať kategóriu“. Odstránenie kategórie prebieha aj s jej potomkami. Kategórie je možné posúvať hore alebo dole

vrámci ich súrodencov. Treba si uvedomiť, že editácia stromu prebieha na strane klienta pomocou javascriptu. Preto pre uloženie zmien je nutné explicitne aktivovať tlačidlo „Uložiť“, ktoré pomocou AJAX odošle reťazec obsahujúci informácie o novej štruktúre kategórii. V prípade, že užívateľ znovu načíta stránku bez uloženia zmien, príde o ne.

2.11.4. Správa vzorov

Práci so vzormi venujem samostatnú časť v tejto kapitole: viď 3.13. Tu len stručne spomeniem ich miesto v aplikácii.

Funkcia vzorov je zjednotenie parametre pre podobné druhy produktov. Tým sa urýchľuje aj tvorba samotných produktov. Parametre produktu prezentuje module ProductParams. V budúcnosti aplikácia ráta s modulom Filter, ktorý rozšíri ich význam pre prezentáciu obchodu a umožní na základe prvkov vzoru filtrovať produkty používajúce daný vzor. Pri súčasnom návrhu vzorov by bolo nutné, aby stránka s kategóriou, ktorá by chcela využívať modul Filter, mala vzťah práve s jedným vzorom produktu. Pričom do tejto kategórie by išlo zaradiť len produkty používajúci rovnaký vzor ako má kategória.

2.11.5. Produkty

Zrejme najpodstatnejšia časť pre elektronický obchod je správa produktov. Preto ani v tejto sekcii som nešetril časom a vytvoril som prostredie na pohodlnú tvorbu produktov pre nenáročných i náročných obchodníkov. Hneď v úvode treba mať na zreteli, že jediným povinným parametrom produktu je jeho názov. Dôvodom bolo nechať na prenajímateľovi tvoriť si vlastnú agendu do akej miery bude poskytovať informácie o produkte. Moduly zobrazujúce časti produktu sú postavené tak, aby nevykresľovali prázdne polia.

Stránka s produktami nám ponúka zoznam produktov. Užívateľ má možnosť úpravy a zmazania produktu. V pravom hornom rohu je odkaz na vytvorenie produktu, ktorý užívatela naviguje k záložkami rozdeleného editora produktu. Záložky síce znížili prehľadnosť validácie vstupov, ale zoskupili logický príbuzné vstupy pre popis vlastností a práce s produktom.

Prvým logickým celkom sú základné informácie. Popri klasických textových poliach prináša neštandardný prístup výber výrobcu. Ak neexistuje žiadny výrobca, tak editor ponúka textový vstup na jeho vytvorenie. Zadaný výrobca bude vytvorený aj v prípade, ak produkt zlyhá pri validácii. V prípade existencie výrobcu aplikácia ponúkne užívateľovi výber medzi textovým poľom pre tvorbu nového alebo rozbaľovacím výberom z už vytvorených výrobcov. Dôležitým prvkom v tejto záložke je výber statusu produktu. Ak je status nastavený na „Skrytý“, produkt sa nebude zobrazovať v prezentácii. V prípade statusu „Zobrazený“ mu v prezentovateľnosti nič nebráni.

Druhou záložkou je logistika, ktorú tvoria vstupy závislé na dátach z časti „Logistika“. V prípade ak prenajímateľ vytvoril aspoň jeden obchod, tak každý produkt si vytvorí vzťah s obchodom alebo skladom. S tým súvisia aj nastavenie základných hodnôt pre dobu dostupnosti a doobjednania tovaru.

Prostredná záložka ponúka pridávanie fotografií k produktu. Jednotlivé fotografie sú ukladané na server asynchrónne pomocou javascriptovej knižnice[19]. Tento spôsob umožňuje rýchlu interakciu klienta a servera, čo vytvára benefit pre

užívateľa v podobe dojmu práce s desktopovou aplikáciou. Nahraté obrázky možno mazať. Pokiaľ nie je produkt reálne vytvorený, resp. nie sú uložené zmeny, tak sú novo nahraté obrázky len v dočasnom úložisku. Pri práci s obrázkami je vhodné využívať v rámci ich mena v url dátum ich uloženia v nejakej číselnej podobe. Takýmto spôsobom predídeme „cacheovaniu“ obrázku v rámci klientovho webového prehliadača, ktoré by spôsobilo v prípade nahratia obrázku s rovnakým menom alebo označením na strane servera zobrazenie už zmazaného obrázku s rovnakou url.

Obsah a prácu záložky Parametre produktu popisujem v časti 3.13. Tvorba produktových vzorov nižšie v tejto kapitole.

Posledná záložka je editor vzťahu medzi kategóriami a produktmi. Nástroj vykreslí kategórie v stromovej štruktúre vo formáte: zaškrtávacie pole a názov kategórie. Zaškrtnutím políčka sa priradí produkt do kategórie. Aby sa predišlo situácii, že by sa produkt nachádzal nesúvislo v rámci jednej cesty z listu do koreňa v strome kategórií, je možné zaškrtnúť len listy stromu kategórie. Po zaškrtnutí automaticky pomocou javascriptu aplikácia označí cestu po predkoch až k najvyššej kategórii. Odobratie produktu funguje presne opačne. Zmeny vykonané na strome kategórií sa prejavia až po uložení produktu.

2.11.6. Objednávky

Táto sekcia je z pohľadu funkcií oproti ostatným chudobná. Obsahuje stránku s objednávkami.

2.11.7. Správa Webu

Ide o najrozsiahlejšiu časť aplikácie. Jej funkcie a práca s ňou sú popísané v nasledujúcej časti.

2.12. Editácia vzhľadu

Otázka je, akú mieru voľnosti zvoliť pre dostatočnú rozlíšiteľnosť obchodov. Niektoré spomenuté generátory elektronických obchodov umožňujú meniť len štýly niektorých prvkov. Napríklad pozadie hlavičky, stránky a fonty textu. Iné umožňujú meniť aj rozloženie prvkov stránky a to buď jednotlivo prechodom po každej stránke prezentácie (prechod v editovacom móde cez každú stránku produktu, katalógu – vhodné len pre veľmi malé obchody), alebo vytvorením schémy, ktorú následne aplikujú na jednotlivé stránky.

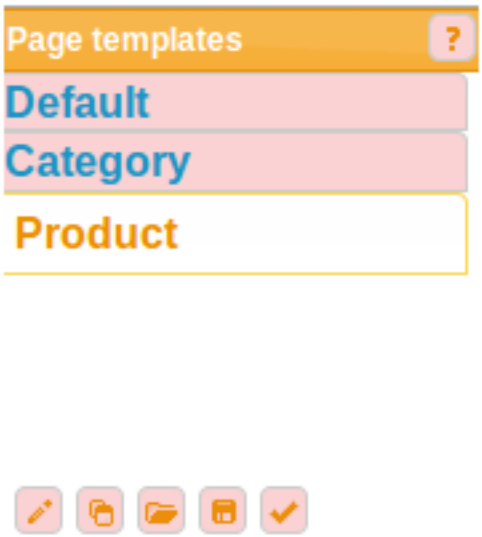
Touto cestou som sa vydal aj ja. Projekt ponúka možnosť meniť jednotlivé poradie modulov na stránke a ich umiestnenie vrámci blokov. Samotné stránky sú rozdelené na bloky. Nie každá stránka musí zobrazovať všetky bloky. Užívateľ to má vo svojej moci, pričom okrem povinného obsahového bloku má na výber ďalších šesť. Ich poradie je nemenné. Takto vytvorené schémy vzhľadu asociuje s jednotlivými stránkami obchodu podľa ich typu, pričom riešenie rozoznáva tri druhy stránok. Stránky detailu produktu, stránky katalógu produktov a systémové stránky. Popísaným spôsobom dosiahneme najväčšiu možnosť slobody tvorby vzhľadu pri najmenšej námahe.

2.12.1. Editor

Tento krátky úvod nás oboznámil s reáliami, okolo ktorých sa budeme v sekcii Webu a jeho vzhľadu pohybovať. Teraz sa ponoríme do úvodnej stránky sekcie Web. Tú tvorí navigácia do jednotlivých podsekcií a prepracovaný editor. Ako bolo už v úvode naznačené, needitujeme priamo jednotlivé stránky obchodu, ale akési schémy (vzory).

Na ľavej strane pod navigáciou je paleta nástrojov. Prvý box s titulkom „Page templates“ obsahuje schémy stránok (Obrázok č. 10). Schéma, ktorú práve editujeme, je zvýraznená. Existujú tri nezmazateľné schémy „Default“, „Category“ a „Product“. Sú to základné schémy a pre prácu so vzhľadom prezentácie obchodu reprezentujú najširší kontajner, v ktorom sú uložené ostatné prvky (pre predstavu v rámci html reprezentuje takýto kontajner tag <body>). V spodnej časti boxu máme nástroje, ktoré aplikujú zmeny na schéme. V poradí zľava doprava podľa Obrázka č. 10 máme:

- Ikonu otvárajúcu dialógové okno s editorom štýlov.
- Tlačítko otvárajúce dialógové okno s nástrojom na klonovanie štýlov medzi vzormi stránok a ich aktívnymi blokmi.
- Otvára možnosti zvoliť aktuálnu verziu vzoru stránky a verziu vzoru modulov.
- Ukladá v prípade neuložených zmien novú verziu podľa typu zmeny (buď verziu vzoru stránky alebo vzoru modulov).
- Publikuje aktuálnu verziu vzoru stránky a modulov.

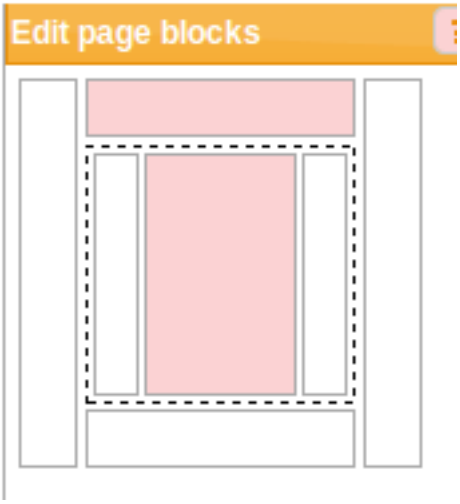


(Obrázok č. 10 – panel nástrojov stránkových vzorov)

Pod boxom so vzormi stránok sa nachádza box reprezentujúci bloky na stránky. Aplikácia poskytuje úplnú voľnosť a teda je len na užívateľovi a jeho schopnostiach, akú štruktúru bloky vytvoria. Prípadne môže použiť niektorý verejne dostupný vzor v rámci celého systému. Kliknutím sa aktivuje blok v tomto prvku (kliknutím na tu symbolickú mapku blokov na stránke). V prípade, že sa blok nenachádza na stránke, zobrazí sa voľba pre jeho pridanie (zobrazenie). Druhým prípadom je zameranie bloku umiestneného vo vzore stránky. Tento zobrazí tlačidlo na otvorenie editoru s jeho štýlmi a v prípade, ak sa nejedná o Obsahový blok,

sprístupní aj tlačítko na odstránenie bloku zo stránky. Poradie vykreslenia blokov generátorom prezentácie je vzhľadom na mapku na Obrázku č. 11 nasledovné:

1. Prvý je ľavý vonkajší blok.
2. Pravý vonkajší blok.
3. Blok zobrazujúci hlavičku stránky na obrázku.
4. Ľavý vnútorný blok.
5. Pravý vnútorný blok.
6. Obsahový (centrálny) blok.
7. Blok reprezentujúci pätičku stránky.



(Obrázok č. 11 – mapa blokov na stránke)

Posledným blokom na ľavej strane editora je kontajner vzorov modulov. Vzory modulov sú spomenuté v práci prvý krat, preto je vhodné odlíšiť ich od samotných modulov. Vzor modulu je objekt, ktorý zabezpečuje obsah, nastavenia a dostupnosť modulu v rámci typu stránky. Modul je umiestnením Vzoru modulu v rámci bloku stránky. Okrem obdobných tlačidiel spomenutých vyššie majú tlačidlá na vytvorenie modulu v bloku stránky. Niektoré vzory modulov disponujú aj tlačidlami otvárajúcimi dialógové okno so správou obsahových prvkov alebo nastavení ovplyvňujúcich správanie modulu.(Obrázok č. 12 – časť kontajneru so Vzormi modulov)

Pravú stranu editora tvorí akési plátno, ktoré verne umiestňuje prvky stránky a zobrazuje ich vzhľad. Jednotlivé prvky na stránke, teda najmä moduly, sú aktívne. Po nabehnutí myšou nad modul sa rozbalí jeho ponuka funkcií. Popri štandardnej správe štýlov modulu ponúka tlačidlá na posúvanie modulu hore a dole v rámci bloku, premiestnenie do iného bloku či odstránenie modulu zo stránky. Jediný modul, s ktorým nie je možné manipulovať je vzor modulu System. Je umiestnený v obsahovom bloku. Dôvodom tejto reštrikcie je spôsob implementácie generátora štruktúry stránky v prezentácii obchodu(štruktúrou sa v tomto prípade rozloženie modulov, blokov a zakomponovanie systémového obsahu prezentácii).
Medzi zaujímavé vlastnosti plátna patrí presúvanie modulov pomocou „drag and drop“ mechanizmu v rámci bloku i medzi blokmi, alebo obdobne pri pridávaní modulov z kontajneru vzorov modulov.

Neopomenuteľnou funkciou plátna je aj schopnosť zobrazovať buď ukážkový obsah modulov (pomocou tlačidla „Zobraziť ukážkový obsah“), alebo obsah s reálnymi dátami náhodne vybranými zo systému („Zobraziť generovaný obsah“). Plátno je reprezentované html tagom s atribútom identifikátoru „canvas“. Pred použitím je očistený od okolitých štýlov stránky, aby sa zobrazovali v tagoch umiestnených v ňom len štýly definované štruktúry prezentácie obchodu. Druhou alternatívou, ktorá by zabezpečila úplnú čistotu zobrazovania je spraviť plátno ako tag iFrame a vložiť doň generovanú editovateľnú stránku. S týmto riešením som experimentoval. Toto riešenie je čiastočne implementované, no upustilo sa od neho

pre zlé možnosti testovania a nedostatočnú podpory zo strany jQuery pre prácu medzi panelom nástrojov a štruktúrou stránky.

2.12.2. Editor štýlov

Pri použití editoru štýlov je dobré si uvedomiť, ktorý prvok čo reprezentuje vrámci stránky:

- Vzor stránky reprezentuje hlavný kontajner, teda vonkajší okraj.
- Blok je kontajnerom modulov a teda tvorí hranicu medzi stránkou a modulom.
- Vzor modulu. Štýly definované vo vzore budú zobrazené aj v module. No ak v module definujeme rovnaký štýl, bude zobrazovaný štýl modulu (modul je v rámci prvkov stránky špecifickejší). Vzor modulu definuje obsah a vlastnosti modulu.
- Modul určuje pozíciu vykreslenia vzoru modulu. Jeho vzhľad je upravovateľný.

Editor okrem zmeny štýlov spomenutých prvkov umožňuje meniť aj triedy definované vo vnútri ich kontajnerov. Zoznam upraviteľných tried si nesie každý sám. Editor dokonca umožňuje vzorom modulov editovať kontajnery ich špecifických obsahových prvkov.

Editácia prebieha v dialógovom okne, kde na ľavej strane sú ciele, na ktoré sa štýly aplikujú (kliknutím na cieľ sa zvýraznia v palete dotknuté prvky) a na pravej strane sú nástroje na zmenu štýlov. Textové vstupy po vyplnení nasadíme po stlačení klávesy Enter. Výberové i rozbaľovacie možnosti sa aplikujú ihneď. Editor ponúka interaktívne nástroje na výber farby a pridanie obrázku pozadia. Obrázky sa ukladajú na jedno miesto a sú dostupné pre všetky ciele zmeny štýlov podporujúce zmenu obrázku pozadia. Po aplikácii štýlov sa automatický vytvára nová verzia vzoru stránky alebo vzorov modulov.

2.12.3. Vzory modulov

Ako prvé budú popísané základné vzory modulov. Tie sú dostupné pre každý typ vzoru stránky.

- **„BreadCrumbs“** vzor poskytuje horizontálnu navigáciu. Odkazy sú vytvárané automaticky podľa hĺbky vnorenia v rámci obchodu.
- **„CategoryMenu“** vypisuje štruktúrovaným spôsobom odkazy na kategórie obchodu. Dovoľuje nastaviť jednoduchý vertikálny výpis (samozrejme existuje možnosť ho upraviť) alebo výpis s možnosťou rolovania podstromov kategórii (tzv. rozbaľovanie, zbaľovanie odkazov).
- **„MainHeadline“** jednoduchý vzor s editovateľným obsahom. Umožňuje na stránku vložiť text zabalený v tagu <h1>.
- **„ContentHeadline“** jednoduchý vzor s editovateľným obsahom. Umožňuje na stránku vložiť text zabalený v tagu <h2>.
- **„Menu“** - tento vzor s krátkym menom disponuje s pomerne mohutným aparátom. Má vlastný editor obsahu. Vytvára v ňom stromové štruktúry s odkazmi. Tiež má nastavenie umožňujúce vytvoriť dynamické horizontálne i

vertikálne menu, zrolovateľné vertikálne menu alebo jednoduchý zoznam odkazov. Tento prvok si nesie aj zoznam položiek menu pre editor štýlov. Tak možno dosiahnuť takmer ľubovoľný vzhľad aj na úrovni tlačidiel menu.
- **„Cart“** Modulu „Cart“, ktorý zobrazuje informácie o cene tovaru v košíku, jeho počte, či odkaz na stránku košíka.
- **„QuickLogin“** obsahuje prihlasovací formulár pre zákazníka obchodu a odkaz na stránku s registráciou.

Nasledujúce vzory sú prístupné len pre vzory stránok kategórií.
- **„ContentHeadline“** umožňuje na stránku vložiť názov kategórie zabalený v tagu <h2>.
- **„ProductList“** vzor ponúka stránkované náhľady produktov s možnosťou radiť ich podľa ceny. Obsahuje sadu tried, ktoré nám pomáhajú rozlíšiť párny a nepárny náhľad.

Posledné vzory majú úlohu prezentovať obsah na stránkach detailov produktov.
- **„ProductTitle“** umožňuje na stránku vložiť názov produktu zabalený v tagu <h2>.
- **„ProductParams“** vykresľuje dvojicu názov parametru a jeho hodnota.
- **„ProductLogistic“** znázorňuje informácie o logistike produktu, jeho dostupnosť a iné.
- **„ProductInfo“** obsahuje základné informácie produktu ako dlhý a krátky popis, jeho cena a iné.
- **„Gallery“** zobrazuje obrázky produktu. Tento modul sa do budúcnosti plánuje rozšíriť o nastavenia, kde by sa dali vybrať typy prezentácie obrázkov.

2.12.4. Zdieľanie vzorov

Vzory, ktoré sú vytvárané, je možné zabaliť do takzvaného webového vzoru (šablóny webu) a následne zdieľať buď medzi vlastnými obchodmi, alebo so širokou verejnosťou. Na vytvorenie šablóny slúži dialóg zobrazený po stlačení odkazu „Save all as web template“ v Editore. Balíčku je potrebné zadať meno (nemusí byť unikátne) a existuje možnosť priradiť k nemu aj obrázky. Stlačením tlačítka „Save“ sa balíček odošle webovej službe sprostredkovanej aplikáciou eGen, ktorá šablónu ďalej spracuje.

Odkazom „Web templates“ naviguje aplikáciu na stránku s verejnými a súkromnými balíčkami vzorov. V prípade ak užívateľ rozhodne niektorú aplikovať, vymažú sa jeho verzie vzorov, vzory modulov i stránok. Nová šablóna nemá vplyv na vzhľad prezentácie kým sa nepublikuje vzory stránok v editore.

2.12.5. Ostantné podsekcie

Ostatné podsekcie tvoria prehľady o vytvorených vzoroch stránok (odkaz „Page Templates“) a modulov (odkaz „Module controls“) . Sekcia „Page Templates a Site map“ umožňuje asociovať vzor stránky, konkrétnej stránke.

2.13. Tvorba produktových vzorov

Pre vytvorenie samotného vzoru stačí zadať jeho názov a aspoň jeden parameter. Počet parametrov nie je explicitne zhora ohraničený. Parametrom vzoru sa rozumie trojica - názov parametra, jeho typ a základná hodnota. Typ parametra určuje akými hodnotami parameter môže disponovať. Tieto hodnoty má možnosť užívateľ následne vybrať pri tvorbe produktu v záložke vzor. Po výbere vzoru sa odošle pomocou javascriptu dotaz na server. Ten vráti buď správu z chybou alebo formulár zostavený podľa parametrov vzoru. Jednotlivé položky formuláru obsahujú možnosti pre vytvorenie parametru produktu (Parameter produktu je dvojica názov parametru – rovnaký ako názov parametru vzoru a hodnota). Pozrime sa teraz bližšie na samotné typy parametru vzoru a nástroje, ktoré sa budú ponúkať pri tvorbe produktu:

- Prvým typom na výber v dialógovom okne tvorby parametru vzoru je „Yes or No“. Pri tvorbe produktu sa ponúkne výberový vstup (radio button) medzi hodnotami Áno a Nie s prednastavenou hodnotou podľa základnej hodnoty parametru vzoru.

Príklad: Vzor „notebook“ s parametrom vzoru X(„WIFI“, „Ano alebo Nie“, „ANO“). Následne pri tvorbe konkrétneho produktu ASUS a6km (notebook) sa vyberie vzor „notebook“, zobrazí sa formulár s výberom WIFI – Ano alebo Nie. Po kliknutí na „NIE“ a uložení zmien bude produkt disponovať parametrom produktu Y(„WIFI“, „NIE“).

- Nasledujúci typ je „Miera“, ktorého hodnota pozostáva z čísla a typu miery, ktorá sa vyberá pri jeho tvorbe. Na výber má „m“ - metre, „m2“ metre štvorcové a „m3“ metre kubické. Vo formulári v produkte je textový vstup určený na číselnú hodnotu, ktorá sa validuje pri uložení produktu.
- „Váha“ je opäť typ s číselnou hodnotou a typom miery. V tomto prípade sa vyberá z „kg“ pre kilogramy, „t“ pre tony, „g“ gramy a „lbs“ pre libry. Hodnota môže obsahovať desatinné číslo a je tiež validovaná pri uložení produktu.
- Zaujímavým typom je „Množina“. Pri tvorbe tohto parametru sa vytvára aj množina možných hodnôt. Hodnoty sú textové. Formulár v editore produktu ponúkne rozbaľovací box (select box) s hodnotami množiny.
- Jednoduchým o to pre slobodu tvorby parametrov dôležitejším je typ „Text“. Jeho názov prezrádza, že jeho hodnotu bude tvoriť ľubovolný text, ktorý budeme môcť pri tvorbe produktu zmeniť.
- Posledným typom je „celé číslo“, s textový vstupom validovaným na celé číslo.

Vzory nie je možné po uložení upravovať. Je možné ich odstrániť, v takom prípade si produkt používajúci odstránený vzor ponechá parametre od neho odvedené do najbližšej úpravy.

2.14. Nastavenia obchodu

Nastavenia obchodu sú v rannej verzii a obsahujú napríklad odkaz na úpravu licenčných podmienok obchodu alebo zmenu zobrazovanej meny obchodu. Prístup k nastaveniam má len majiteľ obchodu.

2.15. Základné použitie prezentácie obchodu

Aplikácia vykonáva už vyššie spomenuté funkcie pri elektronických obchodoch. Na to, aby zákazník mohol ísť k pokladni, musí byť registrovaný. Preto pri tvorbe vzhľadu je dôležité klásť dôraz na to aby mal vždy na dosah moduly sprístupňujúce prihlasovanie do systému, odkaz na registráciu, či stránku košíka odkiaľ vedie odkaz na stránky pokladni.

3. Implementácia

V tejto kapitole sú popísané vybrané časti implementácie v rámci celého systému. Jedná sa o celky, na ktorých stojí Elektronický generátor obchodov. Nachádza sa v nej aj popis kľúčových tried.

3.1. Databázová vrstva

Už z popísaného použitia systému je jasné o aké rozsiahle databázové schémy jednotlivých aplikácii ide, preto sú spomenuté aj v Prílohe B. Zvedavý čitateľ nájde sql skripty pre aplikáciu eGen ktoré sú umiestnené v rámci projektu eGen.Sql a pre aplikácie obchodu je to projekt EshopInstance.Sql. Nachádzajú sa na priloženom nosiči v adresári s projektom Generátor elektronických obchodov.

3.2. Autorizácia

ASP.NET ponúka pohodlne definovať pravidlá a úroveň prístupu užívateľov k jednotlivým stránkam aplikácie v konfiguračnom súbore. Príkladom je prístup k stránke EshopInstance.aspx, ktorá je určená administrátorom aplikácie nasledovne:

```
….
<location path="EshopList.aspx">
    <system.web>
        <authorization>
            <allow roles="admin"/>
            <deny users="?"/>
            <deny users="*"/>
        </authorization>
    </system.web>
</location>
….
```

Atribut *roles* určuje nutné overenie užívateľa na prístup k stránke. Takýmto spôsobom za nás ASP.NET vyrieši explicitnú autorizáciu z našej strany počas životného cyklu stránky. Otázkou je akým spôsobom sa .Net framework dozvie v akom vzťahu je aktuálny užívateľ s požadovanou rolou? Odpoveďou na ňu je abstraktná trieda *System.Web.Security.RoleProvider.* Samotný framework obsahuje niekoľko jej implementácii[20]. Bohužiaľ, žiadna nie je vhodná pre schému databázy aplikácie eGen. Riešením sa stala vlastná implementácia. Oživenie tejto abstraktnej triedy je na pleciach triedy *eGen.Services.RoleProvider.* Aby framework vedel, s čím má pracovať a akú triedu má inštanciovať, je nutné dať mu to vedieť prostredníctvom pár riadkov v konfiguračnom súbore:

```
….
<roleManager enabled="true" cacheRolesInCookie="true" defaultProvider="RoleProvider">
    <providers>
        <clear/>
        <add name="RoleProvider" type="eGen.Services.RoleProvider"
applicationName="/eGen"/>
    </providers>
</roleManager>
….
```

Pre porovnanie je vhodné uviesť aj riešenie autorizácie v aplikáciách EshopInstance a EshopInstance. Administration, ktoré sú postavené na ASP.Net MVC. Nutnosť implementovať *RoleProvider* a aj upovedomenie v tomto prípade na triedu *EmployeeRoleProvider* knižnice EshopInstance.Services frameworku zostalo rovnaké. Kvôli MVC modelu existuje možnosť kontrolovať nielen prístup k jednotlivým kontrolerom, ale aj k ich akciám. Tento luxus ešte zvýrazňuje aspektovo orientovaný prístup, ktorý zvolili autori MVC frameworku [21]. Vložením atribútu Authorize pred metódu reprezentujúcu akciu kontrolóru (v tomto prípade je to akcia Index kontrolóru Employee) sa definuje prístup len pre užívateľa v role majiteľa alebo správcu zamestnancov:

```
….
[Authorize(Roles = RoleService.RoleNameOwner + "," +
RoleService.RoleNamePersonalist)]
public ActionResult Index()
{
   IEmployeeService employeeService = new EmployeeService(this._modelStateWraper,
this._eshopInstanceDBEntities);
   this.ViewData.Model = employeeService.ListEmployees();
   this.ViewData["ownerLogin"] = AppService.OwnerLogin;
   return View();
}
….
```

3.3. Vytvorenie obchodu

V podstate ide o hlavnú operáciu aplikácie eGen. Vytváranie obchodu môže byť iniciované dvomi stránkam,i a to buď Registration.aspx alebo NewEshop.aspx. Obe pracujú s objektom triedy *EshopServices* a volajú metódu *CreateEshop(...).* Tá zabezpečí všetkých potrebných aktérov na vytvorenie obchodu. Prvým je trieda *eGen.Utility.SqlClient* postavená za účelom s práce s projektami eGen.Sql a EshopInstance.Sql, ktoré obsahujú sql skripty potrebné na vytvorenie databázovej vrstvy oboch aplikácii. Pred vytvorením samotnej schémy sa najprv vytvorí databáza nového obchodu:

```
….
sqlClient.CreateDatabase(eshopDbName);
….
```

Následne vytvorí samotnú schému:

```
….
sqlClient.CreateDatabaseSchema(
    eshopDbName
    , EshopService.GetPhysicalPathOfSqlFileMap()
    , EshopService.pathOfEshopSourceSqlDir
);
….
```

Prvý parameter je meno novej databázy zložené z mena databázy aplikácie eGen, reťazca „_eShop_“ a identifikátora obchodu. Druhým je cesta k súboru, ktorý obsahuje zoznam relatívnych ciest k skriptom potrebných na vytvorenie schémy. Tretím parametrom je cesta ku koreňovému adresáru obsahujúcemu samotné skripty. Ukážka obsahu sqlFilesMap.txt:

```
….
# 4. Create modules tables
```

```
/Create Scripts/CreateTables/WebModule/CreateWebModuleCategoryMenuTables.sql
/Create Scripts/CreateTables/WebModule/CreateWebModuleContentHeadlineTables.sql
/Create Scripts/CreateTables/WebModule/CreateWebModuleLogoTables.sql
        …
#
-/Create Scripts/sp_Add/sp_web_AddBlockOnPage.sql
#
        ….
```

SqlClient vyhodnotí riadky začínajúce '#' ako komentáre a ignoruje ich. Ostatné riadky ďalej spracúva. Ak sa cesta k súboru začína '/', tak sa spustí daný skript nad databázou. V prípade znaku '-' sa jedná o viaceré dotazy, ktoré najprv rozdelí (delimitrom je príkaz „GO“) a až potom samostatne odošle databázovému serveru. S pridaním prenajímateľa obchodu do databaźy obchodu ako užívateľa v role „majiteľ“ sa končí jeho úloha.

Štafetu predá triede *eGen.Utility.FileUtility*, ktorá skopíruje vzorové aplikácie obchodu a jeho administrácie do adresárov nového obchodu. Tým je vytvorená jeho fyzická štruktúra.

```
        ….
this.SetEshopConfigFiles(eshopID);
        ….
```

Hore uvedený kód volá metódu triedy *EshopServices* a vytvorí unikátne konfiguračné súbory pre nový obchod. Jedná sa hlavne o zadanie „connection stringov“ pre pripojenie do databázy a údajov do rozšírenia konfigurácie v podobe súboru „release.conf“.

Magický úsek začlenia aplikácii do IIS, obhospodarí trieda *iisClient.* Je nutné najpv vytvoriť virtuálnu cestu obchodu a až následne registrovať obchod ako aplikáciu v ApplicationPool-y eGen-u. Čo je posledným významným krokom k spusteniu služby disitribuovaného obchodu.

3.4. Postup spracovania bežných dotazov v EshopInstance.Administration

Tento odstavec demonštruje rutinný implementačný postup často použitý v MVC aplikáciach EshopInstance a EshopInstance.Administration. Príkladom bude dotaz klienta na zobrazenie zoznamu zamestnancov.

Príbeh aplikačnej vrstvy začína v súbore global.asax.cs, v ktorom trieda *MvcApplicaiton* rozširuje základnú triedu ASP.NET aplikácie, čím nám ponúkne nástroje ASP.NET MVC frameworku. Dôležitá je jej metóda *RegisterRoutes(RouteCollection routes)*, v ktorej sa definuje spôsob, akým sa budú na základe url vytvárať kontrólery, argumenty a volať ich akcie. V tomto príklade URL dotaz na zoznamu zamestnancov bude vyzerať nasledovne : „<doménove meno aplikácie>/Employee“ a jej spracovanie prebehne podľa posledné uvedeného smerovania:

```
        ….
routes.MapRoute(
    "Default",                                  // Route name
    "{controller}/{action}/{id}",               // URL with parameters
    new { controller = "Home", action = "Index", id = "" }  // Parameter defaults
);
        ….
```

Keďže v URL je definovaný len názov kontróleru („controller“) , tak sa na ňom vykoná základná akcia „Index“. Nasledujúce dejstvo sa odohrá v triede kontrólera

EmployeeController dediaceho po triede ApplicationController menného priestoru EshopInstance.Adminstration.Controllers. Akciu reprezentuje jeho metóda Index(). V nej sa vytvorí objekt z vrstvy biznis logiky aplikácie EshopInstance.Services. V tomto prípade je to objekt typu EmployeeService a v konštruktore sa mu predá objekt pre potreby validácie a adaptér k modelom databázovej vrstvy.

```
….
IEmployeeService employeeService = new EmployeeService(this._modelStateWraper, this._eshopInstanceDBEntities);
….
```

Služba poskytne nástroje na získanie potrebných modelov. Samotná trieda sa nedotazuje modelu databázovej vrstvy. Túto záležitosť má na starosti vrstva o úroveň nižšie a reprezentujú ju triedy z menného priestoru EshopInstance.Repository. Triedy majú príznačný názov podľa témy, ktorú spracovávajú, čím sa sprehľadňuje vývoj. Dotazy v tomto prípade nie sú klasické dotazy jazyka SQL, lebo „repository“ vrstva využíva technológiu Linq To Entities, ktorá umožňuje dotazovať nad konceptuálnym modelom EntityFramework-u (v tomto prípade je to spomenutý adaptér k modelom databázovej vrstvy).[26] Technológia poskytuje automaticky transakcie a zabezpečenie proti SQL Injection pri budovaní SQL dotazov. Samotné Linq To Entities nie je určené len pre modely prepojené s databázou, čo tiež umocnilo rozhodnutie pracovať v aplikácii s touto technológiou. Jej použitie v triede EmployeeRepository v metóde GetAll(), ktorá sa volá v metóde ListEmployees vyššej vrsvty, vyzerá nasledovne:

```
….
return (from employee in this.Adapter.EmployeeSet
        select employee).ToList();
….
```

Takýmto spôsobom dostaneme kontroleru požadovaný model zoznamu zamestnancov triedy Ienumerable<Employee>.

Následne sa v rámci vykonávania metódy Index vloží tento model spolu s ďalšími do objektu triedy *ViewData:*

```
….
this.ViewData.Model = employeeService.ListEmployees();
this.ViewData["ownerLogin"] = AppService.OwnerLogin;
….
```

Prvý riadok určuje hlavný model „View-u“ (vykreslovacie časť MVC modelu) a druhý mu poskytuje dodatočný model. Objekt triedy ViewData slúži na presunutie objektov z kontróleru do „View-u“, konkrétny „View-u“ a jeho výstup sa získa zavolaním metódou View(...). Definované sú tri typy: ViewPage, ViewMasterPage a ViewUserControl, ktoré dedia z ASP.NET page (.aspx), master page (.master) a user control (.ascx). [27] Pre dokončenie príkladu je vhodné uviesť, že používa „view“ umiestnený v adresári aplikácie „\Views\Emplyee“ a má meno Index.aspx. Samotná definícia výstupu modelov vyzerá nasledovne:

```
….
<asp:Content ID="ListOfEmployeeContentOutput" ContentPlaceHolderID="MainContent" runat="server">
    <h2>Správa zamestnancov</h2>
….
```

Miesto vykreslenia obsahu v rámci Site.Master schémy, o ktoré sa jedná:

```
….
<table>
    <thead>
      <tr>
        <th>Id</th>
        <th>Meno</th>
```

```
….
        </tr>
      </thead>
      <tbody>
<% foreach (EshopInstance.Data.Employee employee in this.Model)
    { %>
          <tr>
            <td><%= employee.id %></td>
            <td><%= employee.firstName + " " + employee.lastName %></td>
….
            <td class="red"><%= this.Html.ActionLink("Upravit", "Edit", new
{ employeeId = employee.id })%></td>
          </tr>
<% } %>
      </tbody>
</table>
….
```

Uvedený kód demonštuje postupné vykreslenie jednohlavých zamestnancov do tabuľky a tým aj končí z tohto pohľadu spracovanie dotazu na zamestnancov obchodu.

3.5. Validácia na úrovni knižnice EshopInstance.Service

V tejto kapitole sa priblížiakým spôsobom sa presunula validáciu z kontróleru do oddelenej vrstvy. Túto vrstvu reprezentuje knižnica EshopInstance.Services a jej triedy. Dôvodom tejto separácie je sprehľadnenie a odľahčenie tried kontrólerov.

Cieľom je zachovať na úrovni ASP.NET MVC jeho objekt *ModelState* informovaný o valídnosti „nabindovaných“ argumentov akcie bez nutnosti používať triedu *System.Web.Mvc.ModelStateDictionery* v knižnici EshopInstnace.Services. Samotné riešenie je veľmi jednoduché a elegantné. Treba vytvoríť interface *IvalidationDictionary* na úrovni vrstvy služieb.

```
….
public interface IValidationDictionary
{
   void AddError(string key, string errorMessage);
   bool IsValid { get; }
   string ErrorKeyPrefix { get; set; }
}
….
```

Ten sa stane súčasťou jednotlivých tried služieb.

```
….
public class OrderService : IOrderService
{
    private IValidationDictionary _validationDictionary;
    private IOrderRepository _orderRepository;

    public(IValidationDictionary validationDictionary)
    {
        this._validationDictionary = validationDictionary;
        this._orderRepository = new OrderRepository(new EshopInstanceDBEntities());
    }
….
    private bool ValidateLoginMail(string loginMail)
    {
        if (String.IsNullOrEmpty(loginMail))
        {
            this._validationDictionary.AddError(OrderService .LoginMailInputName,
OrderService.ErrorMsgRequiredLoginMail);
            return this._validationDictionary.IsValid;
```

```
        }
            ….
    }
            ….
}
            ….
```

V MVC aplikácii treba vytvoriť triedu ModelStateWrapper implementujúcu IValidationDictionary, ktorá bude obsahovať triedu typu ModelSateDicitonary a cez konštruktor do nej predáme objekt typu ModelState z MVC. Inštancia triedy ModelStateWrapper sa buduje v ranných fázach života v konštruktore triedy ApplicationConntrolle, ktorý je predkom všetkých vytvorených kontrolórov (Controller v MVC), v ktorých sa následne vytvárajú triedy vrstvy služieb s predaním inštancie triedy ModelStateWrapper. [22]

3.6. Hlavné rozhrania a modely menného priestoru *WebGui*

V tejto časti sú popísané niektoré triedy vygenerované EF. Väčšina z nich je ešte následne modifikovaná pomocou partial class, aby implementovala dodatočné rozhrania. Spomenuté triedy sú súčasťou konceptuálneho modelu EF WebAppereanceModel.edmx, ktorý reprezentuje tabuľky uchovávajúce vzhľad aplikácie.

Najprv je potrebné pipísať najpoužívanejšie rozhrania menného priestoru *WebGui* a vlastnosti, ktoré určujú:

- ***IElement*** rozhranie určuje triedy reprezentujúce kontajner na generovanej stránke, ktorý má identifikátor, meno, prefix podľa typu kontajneru (kontajner stránky, modulu, špeciálneho obsahu modulu) a v prípade použitia šablóny webu aj identifikátor a meno triedy kontajnera v rámci štruktúry šablóny.
- ***IHtmlEntityExtension*** rozhranie popisuje prvky v rámci štruktúry stránky (použité najmä pri dotazovaní na elementy pomocou jQuery a pri generovaní pravidiel kaskádových štýlov). Obsahujú teda názov, ktorý je použitý v rámci hodnoty atribútu v html tagu a hodnotu rozlišujúcu typ atribútu, pre ktorý je názov určený. Typ nadobúda hodnoty „.“ pre atribút „class“ a „#“ pre atribút „id“ v tagu.
- ***IHtmlElement*** zjednocuje rozhrania *IElement* a *IHtmlEntityExtension.* Pôvodne malo rozhranie aj iný účel, avšak v priebehu vývoja ho stratilo, no rozhranie zostalo ponechané, kvôli vysokému výskytu v rámci aplikácii a z toho vyplývajúceho nutného veľkého zásahu do kódu na jeho odstránenie.
- ***IStyledEntity*** popisuje triedy s editovateľnými štýlmi. Ponúka model s hodnotami jednotlivých štýlov a zoznam, ktoré podporuje. Ďalej poskytuje zoznam názvov špeciálnych nástrojov pre úpravu štýlov objektu, ktoré podporuje (názvy sú v podstate cesty súborov *UserControlView* obsahujúce daný nástroj).
- ***IStyledClassEntity*** rozhranie má vlastnosti rozhraní *IHtmlEntityExtension* a *IStyledEntity*, k ním pridáva nutnosť niesť objekt triedy *WebClass.* Rozhranie zabezpečuje štýlovateľnú triedu (definovanú v tagu v rámci atributu „class“) v html štruktúre.

- ***IStyledContentEntity*** určuje upravovateľný obsahový prvok v rámci html štruktúry (implementujú ho najmä obsahové triedy reprezentujúce špeciálny obsah vzorov modulov).
- ***IStyledClassesExtension*** definuje nástroje na pridanie novej html triedy vrámci objektu, zmazanie všetkých tried a vrátenie zoznamu tried spojených s objektom.
- ***IStyledContentExtension*** ponúka nástroj na získanie zoznamu špeciálnych obsahových tried.
- ***ISettingExtension*** popisuje triedy umožňujúce meniť svoje nastavenia.

To boli základné rozhrania. Teraz zostáva si predstaviť triedy, ktoré reprezentujú štruktúru stránky. Vhodné je začať tou najpodstatnejšiu, ktorej výskyt na stránke je len raz:

- Triede ***WebPage*** je hlavným kontajnerom html štruktúry (tag <body>). Je možné jej upravovať vzhľad a nesie si aj zoznam štýlovateľných tried. Popri tom má vzťah s 1..n blokmi, kde ten jeden nutný blok je vždy obsahový. Trieda má aj zoznam podporovaných vzorov modulov, kde je tiež minimálne jeden a to Systémový modul.
- ***WebClass*** reprezentuje triedy („class“) v rámci html štruktúry.
- Nasledujúcou triedov je ***WebBlock***, ktorá ma obdobné vlastnosti ako trieda WebPage až na zoznam podporovaných vzorov modulov.
- ***WebModuleControl*** je predkom všetkých tried reprezentujúcich jednotlivé vzory modulov. Ich spoločnými vlastnosťami sú modifikovateľnosť štýlovateľných tried. Ostatné vlastnosti si určujú odvodené triedy, ktoré sú vytvárané už na úrovni EF, ktorý určuje ich typ, podľa parametru „type“ pri záznamoch vzorov modulov v tabuľke.
- ***WebPageBlock*** triedy reprezentujú konkrétne bloky umiestnené v rámci stránky. Nesú rovnaký zoznam editovateľných tried ako triedy WebBlock, s ktorými sú asociované. Tiež sú nositeľmi modulov umiestnených v rámci ich kontajnera v html štruktúre.
- Posledne uvedenou triedou je ***WebPageBlockControl***, ktorá reprezentuje moduly na stránke, resp. hovorí o umiestnení obsahu vzoru modulu vrámci bloku stránky.

3.7. Javascript za Editorom štýlov

Javascript v zastúpení najmä knižnici jQuery hrá v celom projekte veľmi veľkú rolu. V Editore vzhľadu sa snáď ani nedá nájsť aktivitu, za ktorou by sa neskrýval.

avascript nepozná klasickú dedičnosť, no pri tvorbe aktívnych prvkov grafického rozhrania Editoru štýlov by sa náramne hodila. Jednotlivé nástroje majú podobné vlastnosti. Buď obsahujú textové vstupy, výberové tlačítka, alebo objekt na zmenu farby. Len málo nástrojov si vyžadovalo špeciálny prístup ako v prípade štýlu „background-image“ na zmenu pozadia, ktorý riešil nahratie obrázka (v prípade obrázkov by bolo vhodné rozšíriť aplikáciu o správu súborov).

Preto bolo použité prototypovanie na simulovanie klasického dedenia. Nie je to moc vhodné pre javascript, ale sprehľadnilo a urýchlilo to vývoj.[22]

Výsledok možno badať v súbore app-ui-canvas-edit-styles-tool.js, kde sa definujú základné nástroje pre editáciu štýlov a app-ui-canvas-edit-styles-tool-padding.js, kde sa iba inicializujú špecifické vlastnosti pre nástroj na úpravu štýlu odsadenia („padding“).

Ukážka app-ui-canvas-edit-styles-tool-padding.js:

```
….
function CanvasEditStyleToolPadding($context, pushStyleType) {
  this.ini($context);

  this.setContext($context);
  this.setStyle(pushStyleType);
  this.setOptionFixStyleValues(new Array("inherit"));
  this.setThemeDefaultValue("0px");

  this._optionCustomPx = $(".app-ui-canvas-style-edit-tool-custom-px", this.getContext());
    this._optionCustomPercentage = $(".app-ui-canvas-style-edit-tool-custom-percentage",
this.getContext());
};
….
```

Dedenie po nástroji na úpravu šírky:

```
….
CanvasEditStyleToolPadding.inherits(CanvasEditStyleToolWidth);
….
```

Samotný Editor štýlov ponúka zmenu všetkých dostupných kaskadových štýlov verzie 2.0. [28] No tento prístup je vhodný pre náročnejších užívateľov. Preto by bolo vhodné naprogramovať aj odľahčené rozhranie editora. Postup by bolo vytvoriť rovnakým postupom jednoduchšie nástroje a dať im príznak zaradenia určujúci umiestenie v základnej alebo rozšírenej úpravy štýlov.

3.8. Publikovanie vzoru stránky

Po zadaní dotazu na publikovanie vzoru stránky je nutné vytvoriť súbor s kaskádovými štýlmi, predať ho prezentácii obchodu a zapísať dáta z verzovaného vzoru stránky do databázy, keďže prezentácia sa na štruktúru dotazuje.

Pre obe úlohy je nutné vyzdvihnúť aktuálnu verziu vzoru stránky:

```
….
if (HttpContext.Application[WebController.ControllerPrefix +
WebController.ActionSavePageTimeStamp + pageId] == null)
{
….
```

Pri prístupe do Editoru pri zostavovaní štruktúry stránky sa ukladá jej aktuálnu verzia pre zrýchlenie práce s ňou do objektu triedy HttpContext. Pri vyzdvihnutí aktuálnej verzie sa najprv overí, či sa tam nachádza.

```
….
pageVersion=
webVersionService.getLastRevision(WebVersionService.getVersionIdOfWebPage(pageId));

….
```

V prípade, že sa tam nenachádza, prichádza na rad úložisko verzií, kde sa vyhľadá aktuálna verzia.

```
….
// is not versioned create version
if (pageVersion == null)
```

```
                {
                    page = webPageService.getByIdAndPrepareForEditation(pageId);
                    pageVersion = this.addPageRevision(page, "INI - First revision of page id - " +
page.id);
                }
                ….
```

Ak nebola stránka ešte ani verzovaná, je potrebné zostaviť štruktúru z DB a inicializačne ju verzovať. Tento stav by nemal nastať pre publikáciu stránky.

```
                ….
                this.setPageVersionToHttpContext(pageVersion);
            }
            else
            {
                pageVersion                                                    =
(IVersionData)HttpContext.Application[WebController.ControllerPrefix                +
WebController.ActionSavePageTimeStamp + pageId];
            }
                ….
```

Následne sa zostrojí druhý objekt - aktuálne publikovaná verziu na základe dát z DB. Dôvodom je, že serializovaná verzia používaná v Editore vzhľadu, nemá pripojený databázový kontext v rámci entity objektov. EF síce ponúka pripojiť kontext, no s hranicami na konkrétnu inštanciu a jej hodnotové atribúty. Asociované pod-objekty nebudú pripojené, dokonca odstráni z novo pripojeného objektu ich referencie. Takéto správanie bolo limitujúce, keďže jednotlivé vzťahy medzi objektami reprezentujú rozloženie prvkov na stránke a týmto prístupom by sa táto informácia stratila. Riešením je aplikovať zmeny z bez kontextového modulu na model pripojený k EF.

Po aplikovaní zmien sa tieto zmeny uložia a ďalším prechodom cez modely reprezentujúce štruktúru stránky s aktualizovanými identifikátormi sa zostaví príslušný súbor kaskádových štýlov. Posledným krokom je aktualizácia zoznamu už vygenerovaných súborov (vrátane odkazu na štýly šablóny), ktorý sa nachádza v súbore „/Content/published-page-ui-all.css“ link-ovanom prezentáciou obchodu.

3.9. Generovanie prezentácie

Pre generovanie stránok prezentácie sú dôležité dva faktory. Prvý: generátor musí zabezpečiť modely nutné pre daný typ stránky. Pre stránku so zoznamom produktov je to zoznam modelov produktov, či model príslušnej kategórie. S nimi ďalej pracujú vzory modulov. Druhým faktorom je spracovať vzor stránky. To nie je nič zložité. Najprv si v vďaka dátam v tabuľke vypýta aktívne bloky na stránke. Potom zoznamy ich modulov. Pre Obsahový blok rozdelí moduly na dve skupiny. Jednu, ktorá sa nachádza pred Systémovým modulom a druhú, ktorá sa nachádza po ňom.

Vykresľovanie prebieha v Master.Site pomocou volania „PartialControlsViews“ najprv pre bloky, potom vnútri blokov pre moduly.

Jediné prerušenie generovania obsahu predstavuje vykreslenie Obsahového bloku. Ten svoj kontajner má napísaný priamo v Master.Site a iba ho obohacuje o svoju html identitu:

```
                ….
            <div id="<%= this.ViewData["appOutputBlockNameContent"] %>" class="app-block app-
container-center app-container-center-inner <%= this.ViewData["appOutputBlockTplNameContent"] %>"><%
            if (this.ViewData["appOutputModulesContentPartTop"] != null)
            {
```

```
                    // output of top content modules
                    this.Html.RenderPartial("PartialControlsViews\\Output\\Modules",
(List<WebPageBlockControl>)this.ViewData["appOutputModulesContentPartTop"]);
                }
                    ….
```

Tu je miesto pre systémové správy (neskôr je v pláne fixovaný výstup nahradiť modulom)

```
                    ….
                // app messages output
                this.Html.RenderPartial(ApplicationController.appMessagesPartialControlViewPath);
                    ….
```

Potom sú vykreslené moduly umiestnené nad Systémovým modulom.

```
                    ….
                // system module output
                %>
                <div id="<%= this.ViewData["appOutputModuleSystemName"] %>" class="app-module <
%= this.ViewData["appOutputModuleControlSystemName"] %> <%=
this.ViewData["appOutputModuleControlSystemTplName"] %>">
                    ….
```

ContentPlaceHolder je namiesto systémového modulu a umožňuje vykresľovať obsah systémovým stránkam, ako registrácia užívateľa, pokladne, košík i zákaznícke rozhranie.

```
                    ….
                  <asp:ContentPlaceHolder ID="SystemModuleOutput" runat="server" />
                </div>
                <%
               ….
```

V prípade existencie sa vykreslia zvyšné moduly:

```
                    ….
                if (this.ViewData["appOutputModulesContentPartBottom"] != null)
                {
                   // output of bottom contetn modules
                   this.Html.RenderPartial("PartialControlsViews\\Output\\Modules",
(List<WebPageBlockControl>)this.ViewData["appOutputModulesContentPartBottom"]);
                }
                // END content block
                %>
                    ….
```

3.10. Webová služba aplikácie eGen a webové šablóny

Na vytvorenie reprezentácie šablony webu je potrebná trieda zložená zo zoznamu objektov tried *WebPage*. Všetky si nesú informácie ako vyzerá vzor stránky, ktorý reprezentujú. Medzi ne patrí zoznam objektov triedy *WebPageBlock*. Tieto objekty reprezentujúce aktívne bloky na stránke ukrývajú zoznam tried *WebPageBlockControl*. Šablóna nutne potrebuje vedieť aké vzory modulov boli použité. Takže k už vymenovaným objektom sa pripočíta zoznam objektov tried *WebControl*. Každý so spomenutých objektov nesie informácie o svojich štýloch, o štýloch tried, alebo obsahu a nastaveniach. Vymenované triedy určujú vzhľad a štruktúru webu obchodu a to isté má za úlohu aj šablóna webu. Okrem spomenutých

vlastností od nej potrebuje poznať jej tvorcu (majiteľ obchodu), reštrikcie (či je verejná alebo súkromná).

Vyššie popísané vlastnosti reprezentuje trieda *WebTemplate*. Jej prvotný návrh pracoval s triedami *List<WebPage>* (zoznam vzorov stránok) a *List<WebModuleControl>* (zoznam vzorov modulov), avšak pre reálne použitie vo webovej službe sa ukázalo riešenie ako nevhodné. Webové služby v ASP.NET umožňujú prijímať aj odosielať aj iné ako preddefinované typy. Naivný prístup bol použiť teda triedy z knižnici EshopInstance.Data aj na strane aplikácie eGen vo webovej službe. No po pridaní služby do projektu EshopInstance.Administration, Visual studio okrem vygenerovania triedy adaptéru na prácu s webovou službou vytvorilo aj nový menný priestor s triedami webovej služby, čo neumožnilo jednoducho posielať webovej službe objekt triedy *WebTemplate*. Bolo by preto potrebné vytvoriť triedy „premosťujúce“ jednotlivé typy z EshopInstance.Data do tried z menného priestoru webovej služby, čo bolo z časového hľadiska zamietnuté.

Nasledovala teda zmena prístupu k webovej službe, ktorá začala pracovať len s preddefinovanými typmi jazyka c# (string, bool, integer, ...) . Namiesto triedy WebTemplate teda očakávala služba reťazec, kde bol daný objekt triedy WebTemplate aj s pod-objektami serializovaný vo formáte xml. Nástrojmi na serializáciu tried z priestoru WebGui už aplikácia disponovala pri vytváraní verzií vzorov stránok a modulov. Na základe toho bola predpokladaná ľahká realizácia uvedeného postupu, avšak výsledná dĺžka reťazcov zapríčinila neefektívnu komunikáciu.

Z časového hladiska bolo najlepším riešením zmena triedy WebTemplate, ktorá nahradila triedu List<WebPage> a List<WebModuleControl> reťazcami so štruktúrami vzorov stránok a modulov vo formáte jSon.

```
….
public interface IWebElementService
{
        ….
    JObject getJSonObject(IElement element);
    void setJSonObjectToElement(JObject jSonElement, IElement element);
    void setJSonObjectToElement(JObject jSonElement, IElement element, bool setStyles);
        ….
}
        ….
```

Toto rozhranie implementuje WebElementService. Pomocou vyššie uvedených metód, ktoré pracujú s rozhraniami z menného priestoru WebGui, stačilo prejsť cez objekty v zoznamoch vzorov stránok a modulov a zostavovať objekty triedy JObject patriace knižnici Newtonsoft.Json, pričom ich štruktúra reprezentovala rozhrania, ktoré implementujú triedy z priestoru WebGui. Knižnica Newtonsoft.Json ponúka serializovať a deserializovať jej objekty do reťazca vo formáte Json. Preto nebolo potrebné vytvárať žiadne nové nástroje na riešenie problému a bolo dosiahnuté zmenšenie predávaného reťazca do služby, keďže jSon štruktúra je „ľahšia“ ako XML a tiež nesie len nevyhnutné informácie.

Predchádzajúci text bol o výmene dát medzi obchodom a centrálnym prvkom. No v prípade predávania šablóny aplikácii eGen dochádza na jej strane i strane obchodu k istým modifikáciám dát a následnému spracovaniu, ktoré prebieha nasledovne.

Na strane obchodu je nutné si uvedomiť, že objekty triedy *WebStyle* nesú len informácie o štýloch, ktoré sa upravili v Editore. No vzhľad plátna i generovaného obchodu je tvorený aj Ssprípadnou použitou šablónou nalinkovaním súboru s jej

štýlmi v hlavičke stránky. Preto pred vytvorením nového objektu *WebTempalte* treba zabezpečiť, aby objekty stránok s rozhraním *IHtmlElement* „podedili“ štýly z použitej šablóny. Potom nič nebráni vytvoriť novú šablónu a podať ju webovej služe.

Po deserializovaní prijatého reťazca inštanciuje aplikácia eGen triedu *WebTemplate.* Jej spracovanie prebieha v triede *WebTemplateService.* Najprv vytvorí o jej základných informáciach záznam v tabuľke šablón, ktorý urýchľuje poskytovanie prehľadu o šablónach bez nutnosti použitia celej šablóny. Následne vytvorí jej adresárovú štruktúru v rámci úložiska i adresára distribujúceho zdroje. Potom prejde jSon štruktúry reprezentujúce objekty z *WebGui* a ich štýly. Dôvodom je získanie obrázkov použitých v štýloch uložených na strane obchodu a následnou zmenu zdroja obrázku pri štýle. Po tejto transformácii a centralizácii zdrojov vytvorí súbory s kaskádovými štýlmi pre prezentáciu obchodu i plátno v Editore vzhľadu.

3.11. Odstraňovanie dát

Vzhľadom na funkčné požiadavky a databázovú schému bolo nutné zabezpečiť rozumnú implementáciu odstránenie modelov z pohľadu užívateľa. Príkladom je model Product reprezentujúci predávaný produkt. Položky objednávky sa odkazujú na stránku jeho detailu. V prípade fyzického odstránenia by vznikol neplatný odkaz. Riešením je mazanie produktov s príznakom „zmazaný“, ktorý umožňuje zobraziť takýto produkt, ale so správou od aplikácie, že nie je v aktuálnej ponuke. Toto riešenie sa aplikuje aj pri iných kritických situáciách (napríklad vzory produktov).

Záver

V práci bol popísaný princíp systémov, ktoré generujú (distribuujú) elektronické obchody, ich obmedzenia a prednosti. So zreteľom na uvedený princíp bol navrhnutý a vytvorený systém aplikácií generujúci elektronické obchody. Vytvorené riešenie spĺňa popísané vlastnosti softvéru ako služby. Obsahuje centrálny prvok, ktorý vytvára jednotlivé obchody. Dokonca distribuuje webové šablóny, ktorých zdieľanie medzi obchodmi s rôznymi prenajímateľmi dodáva jemný komunitný nádych. Prenajímaným obchodom systém umožňuje budovať vlastnú identitu vzhľadu i biznis agendu.

Žiaľ konečný výsledok netvorí „dokonalú“ aplikáciu ponúkajúcu importy a exporty produktov, ktoré by uľahčili prechod už zabehnutých obchodov do systému. Tiež v oblasti správy obsahu zaostáva za komerčnými riešeniami ako sú Webnode, alebo Vltava2000. No na druhú stranu pre zbehlého užívateľa ponúka slobodnejší prístup k úprave vzhľadu. Dôraz bol kladený na možnosť rozširovania ponúkanej funkčnosti. Príkladom sú vzory produktov tvoriace základ načrtnutého vzoru modulu filtra produktov v kategórii. Vzormi produktov nedisponuje žiadne spomenuté komerčné riešenie.

Hlavnými nevýhodami a nedostatkami sú:

- Z dôvodov priblížiť systém a hlavne úpravu vzhľadu aplikáciu širšej skupine užívateľov by bolo vhodné implementovať rozšírenie Editora štýlov o záložku so základnými nástrojmi a záložku s pokročilými nástrojmi, kde by bolo presunuté aktuálne riešenie. Náčrt riešenia bol spomenutý v sekcii 4.7. Spomenuté riešenie je náročne najmä na čas na jeho odladenie a citu na vytvorenie dostatočne užívateľsky prívetivých nástrojov.
- Vhodná by bola aj možnosť ďalšej práce s obrázkami po ich nahratí v rámci Editora vzhľadu. V súčasnosti má v tomto aplikácia svoje limity. Tie by zborilo vytvorenie súborového manažéra s ponukou na zmazanie a nahratie obrázka. Išlo by najmä o programovanie dynamických prvkov správy na strane klienta pomocou javascriptu.
- Odstránenie požiadavky registrácie zákazníka pred vykonaním objednávky. Takúto funkčnosť poskytuje spomenutá Vltava2000. Z pohľadu nášho systému by to chcelo dokončiť implementáciu komunikácie aplikácie so zákazníkom obchodu pomocou e-mailu. Aplikácia by vytvárala pseudo registrovaných užívateľov s prístupom k stave objednávky cez odkaz na aplikáciu v e-maily. Riešenie je skoro hotové, len si vyžaduje odladiť.
- Podpora štatistík predaja a iné pohľady na výkonnosť obchodu pre prenajímateľa. Časovo aj implementačne ide o nenáročnú funkčnosť. Stačilo by vytvoriť pár nových dotazov na dátový model a vytvoriť ich prezentáciu.
- Takisto chýba technická podpora na strane aplikácie eGen v podobe monitorovania použitého miesta obchodmi, či jednotlivými obchodmi, alebo zaznamenávanie dát o počte prístupov na jednotlivé obchody. Prístup vytvorenia takýchto štatistík môže použiť buď zabudované monitorovacie nástroje IIS, alebo nechať dáta zbierať samotný obchod a následne predávať centrálnemu prvku cez webovú službu.
- Skoro štandardom v komerčnej sfére softvéru ako služba sa stalo asociovanie vlastnej domény s aplikáciou. Táto funkčnosť by mala mať v prípadnom rozšírení vysokú prioritu. Riešenie spočíva v nastavení domény pre danú

aplikáciu v IIS a poskytnutie klientovi adresu servera presmerovajúceho doménove meno na aplikáciu obchodu.[34] Danú adresu si klient bude musieť zadať v aplikačnom rozhraní poskytovateľa domény.

- Rozšírenie projektu o jazykové lokalizácie. Dnes podporuje len slovenčinu. Riešenia sú buď použiť nástroje ASP.NET ako „global resources“, alebo vytvoriť vlastnú správu jazykov a textov aplikácie na úrovni databázy.
- Plná podpora prehliadača Internet Explorer. Najmä v Editore vzhľadu neboli všetky funkčnosti testované s IE. Preto ho neuvádzam ani v podporovaných prehliadačoch, avšak zbežným testovaním nevykazoval závažnejšie nedostatky.
- Podpora automatizácie aktualizovania obchodov z centrálneho prvku. Riešenie by nebolo náročné, stačí použiť rovnaké nástroje ako pri tvorbe obchodu a v konfigurácii aplikácii eGen pridať adresár so zdrojom aktualizácii.

Výhodou systému je nezávislosť jednotlivých distribúcií, ktorá znižuje možnosť nefunkčnosti celého systému. Tiež poskytuje priestor na individuálny prístup poskytovateľa, k niektorým odberateľom. Vytvorenie nového obchodu je veľmi rýchla a pohodlná záležitosť. Tak isto aj jeho správa ponúkajúca zaujímavé funkcie a prístupy k užívateľskému prostrediu. Podstatným prvkom systému je odbremenenie prenajímateľa od technickej správy svojho obchodu a dostupnosť v rámci celosvetovej siete Internet. Odberateľ sa tak môže plne venovať svojím zákazníkom a agende obchodu.

Navrhnuté a implementované riešenie sa dá rozšíriť aj inými ako navrhnutými zmenami. Existuje možnosť vybrať sa cestou zakomponovania aplikácie správy obsahu a elektronický obchod nechať len ako doplnkovú aplikáciu.

Jednoznačne je možné prehlásiť, že program ponúka mnoho hľadísk na rozšírenie. Či už sa môžu zakomponovať nové vzory modulov poskytujúce pestrejší obsah, alebo rozšíriť agendu na strane centrálneho prvku. Napriek všetkým problémom pri vývoji snáď projekt splnil svoj účel, ktorým je návrh a vytvorenie generátora elektronických služieb umožňujúceho poskytovať elektronické obchody internetovej populácii dostupnejšie ako v prípade zakúpenia hostingových služieb a následného nasadzovania krabicového elektronického obchodu.

Zoznam použitej literatúry

[1] Kenneth C. Laudon, Carol Guercio Traver
E-commerce : business, technology, society -2nd ed. (ISBN: 0-321-20056-x)

[2] Rob Conery, Scott Hanselman, Phil Haack, Scott Guthrie
Professional ASP.NET MVC 1.0 (ISBN: 978-0-470-38461-9)

[3] Tom Archer, Andrew Whitechapel
Inside C# -2nd ed. (ISBN: 0-7356-1648-5)

[4] Jiří Polák, Vojtěch Merunka, Antonín Carda
Umění systémového návrhu, (grada 2003)

[5] XHTML reference
w3c.org

[6] použité štatistíky
http://epp.eurostat.ec.europa.eu/portal/page/portal/information_society/data/database

[7] APEK tlačová správa
http://www.apek.cz/8477/2193/clanek/na-internetu-stale-dominuje-dobirka/

[8] techtarget
http://searchcloudcomputing.techtarget.com/definition/cloud-computing

[9] wiki Cloud computing
http://en.wikipedia.org/wiki/Cloud_computing

[10] SIIA
http://www.siia.net/estore/pubs/SSB-01.pdf

[11] Gartner group SaaS prieskup trhu
http://www.gartner.com/it/page.jsp?id=1739214&M=6e0e6b7e-2439-4289-b697-863578323245

[12] CRM and SaaS štruktúra trhu
http://www.crmforecast.com/saasresearch.htm

[13] SaaS architktúra
http://www.saas-attack.com/SaaSDesign/SaaSArchitecture/tabid/183/Default.aspx

[14] 3-vrstvová architekúra s Asp.Net
http://msdn.microsoft.com/en-us/library/aa581771.aspx

[15] ADO.NET DataSet
http://msdn.microsoft.com/en-us/library/zb0sdh0b(v=vs.80).aspx

[16] ASP.MVC , EF, N-tier
http://www.codeproject.com/KB/aspnet/ASP_NET_MVC_WITH_EF.aspx

[17] Enity Framework
http://msdn.microsoft.com/en-us/library/aa697427(v=vs.80).aspx

[18] ASP.MVC
http://msdn.microsoft.com/en-us/library/dd394709.aspx

[19] ajaxfileupload.js
http://www.phpletter.com/Our-Projects/AjaxFileUpload/

[20] ASP.NET role manger
http://msdn.microsoft.com/en-us/library/ff647401.aspx

[21] ASP MVC atributes
http://dotnetslackers.com/articles/aspnet/ASP-NET-MVC-2-0-Attributes.aspx

[22] Validácia v service vrstve
http://www.asp.net/mvc/tutorials/validating-with-a-service-layer-cs

[23] Protoypavnie
http://javascript.crockford.com/prototypal.html

[24] Linq to jSon
http://james.newtonking.com/pages/json-net.aspx

[25] ASP.NET MVC pipeline
http://blog.stevensanderson.com/2007/11/20/aspnet-mvc-pipeline-lifecycle/

[26] Linq TO Entities
http://msdn.microsoft.com/cs-cz/library/bb386964.aspx

[27] ASP.NET MVC View and UI Rendering
http://msdn.microsoft.com/en-us/library/dd410123.aspx

[28] w3c css reference
http://www.w3.org/TR/CSS21/

[29] jQuery
www.jquery.com

[30] jQuery UI
www.jquery-ui.com

[31] superfish Menu Jquery plugin
http://users.tpg.com.au/j_birch/plugins/superfish/

[34] Domové jména, DNS, Name Server
http://www.earchiv.cz/a98/a816k180.php3

Zoznam použitých skratiek

EF – Enity Framework

DB – Databáza

Prílohy

Príloha A – Návod na inštaláciu systému

Návod na inštaláciu systému nájdete na priloženom médiju v súbore install.pdf.

Príloha B – Popis databázovej Štruktúry

Popis databázovej štruktúry nájdete v priloženom médiu v súbore popis_schémy.pdf

Príloha C – Obsah priloženého média

Na priloženom médiu CD, sa nachádzajú Prílohy A a B. Zdrojové kódy aplikácie a verzia určená na prevádzku.

Zdrojové kódy sú v adresári \source

Aplikácia určená na prevádzku je v adresári \release